CONSTRAINED MODEL PREDICTIVE CONTROL OF A NONLINEAR AEROSPACE SYSTEM

THESIS

Christopher M. Shearer, B.S.
Captain, USAF

AFIT/GAE/ENY/97D-05

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GAE/ENY/97D-05

CONSTRAINED MODEL PREDICTIVE CONTROL OF A NONLINEAR AEROSPACE SYSTEM

THESIS

Christopher M. Shearer, B.S.
Captain, USAF

AFIT/GAE/ENY/97D-05

Disclaimer

The views expressed in this dissertation are those of the author and do not reflect the official policy or position of the United States Air Force, the Department of Defense, or the United States Government.

AFIT/GAE/ENY/97D-05

CONSTRAINED MODEL PREDICTIVE CONTROL OF A NONLINEAR AEROSPACE SYSTEM

THESIS

Presented to the Faculty of the Graduate School of Engineering of the Air Force Institute of Technology Air University In Partial Fulfillment for the Degree of

Master of Science in Aeronautical Engineering

Christopher M. Shearer, B.S.

Captain, USAF

Air Force Institute of Technology

Wright-Patterson AFB, Ohio

December 1997

Sponsored in part by WL/FIGC

AFIT/GAE/ENY/97D-05

CONSTRAINED MODEL PREDICTIVE CONTROL OF A NONLINEAR AEROSPACE SYSTEM

Christopher M. Shearer, B.S.

Captain, USAF

Approved:

Sharon A. Heise

Capt Sharon Heise
Committee Chairman

2 DEC 97
Date

Brad Liebst

Dr. Brad Liebst
Committee member

2 Dec 97
Date

Stuart Kramer

Lt Col Stuart Kramer
Committee member

2 DEC 97
Date

Acknowledgments

I would like to express special thanks to my advisor, Capt. Sharon A. Heise. Over the 18 month program here at AFIT she has always been available to answer all my questions and misunderstandings. Also my thanks go to Dr. Brad Liebst and Lt. Col Stuart Kramer for their enlightenments which helped me through several hurdles in my studies and this thesis. But my greatest thanks goes to my wife, Lt. Heather L. Shearer. Despite our physical separation, Heather at Little Rock AFB and myself here at Wright Patterson, over the past two and a half years, she has always been there for emotional and physical support. Her support was key in my success here at AFIT.

Table Of Contents

List of Figures

Page

Page

Page

Page

Page

Page

List of Tables

List of Symbols

Symbols

p	Prediction Horizon Length
q	Control Horizon Length
ρ	$\max(p, q)$
r	Optimization Horizon Length
$\dot{h}$	Rate of Climb (ft/sec)
ϕ	Bank Angle (deg)
β	Side Slip Angle (deg)
V	Perturbation Velocity (ft/sec)
u	Perturbation Control Inputs
Δu	Change in Perturbation Control Inputs
s	Desired trajectory
y	Perturbation System Outputs
x_{eq}	Reduced order state equilibrium values
$x_{eq_{all}}$	Full order state equilibrium values
$\hat{x}$	State Estimates
κ	Number of States in Controller
κ_{F-16}	Number of States in Nonlinear F-16 Model
ϵ	Number of System Outputs
ξ	Number of System Inputs
F_{r_i}	Location of Discrete Controller Pole of $A + BF_r$ closest to Unit Circle
L_i	Location of Discrete Controller Pole of $A + LC$ closest to Unit Circle

List of Abbreviations

Abbreviation	Definition
AFIT	Air Force Institute of Technology
MPC	Model Predictive Control
FIR	Finite Impulse Response
GPC	General Predictive Control
SGPC	Stabilized General Predictive Control
SISO	Single Input Single Output
MIMO	Multiple Input Multiple Output
LTI	Linear Time Invariant

Abstract

Recent research efforts have applied the receding horizon Model Predictive Control (MPC) strategy to linearized high performance aerospace systems. The research contained in this thesis used these recent results in order to apply the MPC strategy to a nonlinear high performance aerospace system, specifically an F-16 fighter aircraft model. The model was commanded to follow dynamic trajectories of roll angle and altitude. Further, adaptive constraint techniques were used to improve system tracking.

To accomplish these tasks, code and block diagrams were generated using the commercial software packages of Matlab and Simulink. Numerous simulations were conducted with the goal of achieving realistic aircraft performance. In many cases, to improve system tracking and reduce control input oscillations, rigid mathematical constraints previously used in the MPC strategy were relaxed.

CONSTRAINED MODEL PREDICTIVE CONTROL OF A NONLINEAR AEROSPACE SYSTEM

Chapter 1 - Introduction

1.1 Model Predictive Control

Model Predictive Control (MPC) is a control strategy which optimizes a specified performance index over a set of future inputs to minimize future output deviations from a specified trajectory. MPC typically operates on three receding horizons: an optimization horizon, a control horizon and a prediction horizon. An on-line optimization takes place over the optimization horizon while future control inputs are calculated over the control horizon and future output trajectories are calculated over the prediction horizon. The controller then implements the first control input, discards the rest, and recalculates the next series of control inputs at the next time step. Because of the computational intensity due to the on-line optimization, MPC has traditionally been applied to low-bandwidth processes.

Recent work has focused on applying the MPC strategy to high performance linearized aerospace systems [4] , [11] , [6] . Frequently, MPC formulations which use a stabilizing inner feedback loop have been applied to such systems. The use of a stabilizing inner feedback loop guarantees system tracking within a finite number of time steps for discrete controller poles placed at the origin. Future commanded inputs are generated by optimizing over a reference signal, v, which is the input to the inner feedback loop. This optimization allows for an independent optimization horizon, r, because the optimal plant inputs are no longer directly calculated. Also, the use of the inner stabilizing feedback loop ensures stability through the existence of a monotonically decreasing cost function. The distinct advantage of the MPC strategy over traditional controller designs is its ability to account for

changing system constraints, i.e. actuator limits and rates, as well as output state constraints such as altitude and bank angle.

1.2 Importance of Research

Because the MPC strategy is capable of dealing with changing constraints, it is potentially suited to aerospace systems in which complete or partial actuator failures are possible. With aircraft which are statically unstable or at best neutrally stable, human response may not be sufficient to maintain positive control of an aircraft following an actuator failure. Also, as flight research continues into the use of new combinations of actuators and reduced torsional stiffness wings [10] , designing controllers which provide the greatest control authority using the least control power may be difficult using classical control techniques. The MPC strategy is also suited to flight test work because of its ability to handle changing constraints. In an actual flight test, the MPC controller could potentially act as safety watch dog, preventing the pilot or test from exceeding certain parameters. By predicting these future parameters, the actuator constraints can be modified real time to prevent a dangerous situation before it occurs. MPC can also be coupled with current Pilot Induced Oscillations (PIO) research. By predicting future outputs, pilot inputs can be modified to assist current filter designs in the prevention of PIO.

1.3 Research Objectives

Before MPC can be realized on an actual aerospace system, it must be shown to work well on simulations involving a nonlinear plant. Building upon past work, this thesis examined four areas of use of an MPC controller. The first was to expand the recent work [4] of step responses and demonstrate MPC's ability to handle dynamic trajectories. The second was the application of the MPC strategy to a nonlinear F-16 aircraft model. The third area was the relaxation of rigid mathematical constraints associated with pole placement. And last was the application of adaptive constraints

to improve system tracking and reduce overall required control power. These four areas were researched with an overriding emphasis that simulations should model realistic aircraft performance.

1.4 Thesis Overview

Chapter 2 explains the mathematical derivations of the state space MPC formulation. Deviations from earlier works are noted, and potential tuning parameters are identified. The chapter also addresses the rigid mathematical constraints previously used in MPC studies and offers a heuristic approach to achieve asymptotic stability. Chapter 3 details the F-16 linear and nonlinear aircraft models which were used in this research. The two models as well as the entire MPC block diagram are described in detail. Additionally flight conditions, open loop pole locations and potential problems associated with differences between the two models are identified. Chapter 4 presents the results of various MATLAB simulations which were designed to demonstrate MPC's ability to meet the research objectives. Chapter 5 offers conclusions and recommendations, based upon the results of the research, providing starting points for future research.

Chapter 2 - Review of Literature and Mathematical Development

2.1 General Predictive Control and Stable Generalized Predictive Control

This section covers the basics of two of the more common types of Model Predictive Control. The first is General Predictive Control (GPC) and the second is Stable Generalized Predictive Control (SGPC). The advantages and disadvantages of each are discussed and a suitable method was chosen.

2.1.1 Generalized Predictive Control

GPC's basic concept is to minimize a cost function across separate input and output horizons, where a horizon is an integer number of steps into the future. Over the output horizon, the tracking error,

$$\begin{aligned} e &= y(k) - s(k) \\ y(k) &\equiv \text{System Outputs at Time Step } k \\ s(k) &\equiv \text{Commanded Trajectory at Time Step } k, \end{aligned} \tag{1}$$

is minimized. Over the input horizon the optimization minimizes the required control power to reduce the tracking error, e. To do this, GPC uses an explicit plant model of the form

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) \end{aligned}$$

to predict the plant outputs, y, across the output horizon, N. The control inputs are weighted and optimized over a separate input horizon, N_u. The Single Input Single Output (SISO) expectation cost function is

$$\begin{aligned} J(k) &= E\left\{\sum_{l=1}^{N}[y(k+l) - s(k+l)]^2 + \lambda\sum_{l=1}^{N_u}\Delta u(k+l-1)^2\right\} \\ \Delta u(k) &\equiv u(k) - u(k-1) \end{aligned} \tag{2}$$

where λ is the weighting of the control power. An expectation operator is used because the output, $y(k+l)$, can be contaminated by various noise sources.

There are several advantages of GPC over other controllers. First there is no requirement to know the system closed loop poles and there are no ill effects caused by closely spaced zeros and poles [4] . GPC also allows the use of control input constraints as well as system constraints. These constraints provide the control designer more tuning parameters to improve system tracking. However, GPC's major fault is it offers no guarantee of stability.

2.1.2 Stable Generalized Predictive Control

Stable Generalized Predictive Control (SGPC) builds upon GPC's advantages and adds a stability guarantee. This stability guarantee is realized by the introduction of a stabilizing inner feedback loop based upon the Youla parameterization of all stabilizing controllers [7] .With the introduction of an inner feedback loop, the MPC controller now operates on a reference input $v(k)$. This allows the introduction of a new optimization horizon r over which the cost function is minimized. If the discrete poles of the stabilizing inner feedback loop are chosen to be at the origin, Finite Impulse Response (FIR) behavior is achieved. FIR behavior implies that a system will achieve a steady state value over a finite horizon. When used with a cost function that minimizes tracking error, both stability and a monotonically decreasing cost function are guaranteed provided specific horizon conditions are met [4] . In this thesis the requirement for FIR behavior was relaxed and it was shown that a finite error could be reached over a finite horizon. Because of the ability to reach a finite error and the stability guarantee, the SGPC method was chosen.

2.2 State Space Formulation of MPC

The state space formulation of SGPC was done in a four step process. First the SISO cost function of Equation 2 was rewritten in multiple input multiple output (MIMO) form. Once the cost

function was identified, step two involved developing a stabilizing inner feedback loop. In step three the state prediction matrices were developed, which were used by the optimization routine to predict future outputs, states and inputs. Finally, a MATLAB quadratic optimization routine was applied to the equations developed.

2.2.1 Cost Function

An expansion of the SISO cost function of Equation 2 in MIMO form gives

$$J(k) = \sum_{l=1}^{p} \|y(k+l) - s(k+l)\|^2_{R_y} + \sum_{l=1}^{q} \|\Delta u(k+l-1)\|^2_{R_u} \tag{3}$$

where

$$\begin{aligned} R_y &> 0 \\ R_u &\geq 0 \\ R_y &= R_y^T \\ R_u &= R_u^T. \end{aligned} \tag{4}$$

The selection of R_u to be positive semidefinite implies that infinite control power could be commanded of the system. However, this was avoided as the control inputs were constrained. Due to the stabilizing inner feedback loop a reference input $v(k)$ is introduced. This allows the cost function, $J(k)$, to be minimized over the optimization horizon r at each time step k. The difference of system outputs, y, and setpoint trajectories, s, are weighted by $R_y^{\frac{1}{2}}$ over the prediction horizon, p. Incremental control inputs, Δu, are varied along the control horizon, q, while weighted by the matrix $R_u^{\frac{1}{2}}$.

2.2.2 Stabilization of a LTI Plant using Coprime Factorization

The controller used in this thesis was based upon an inner loop feedback which provided stability of the plant, G. Given that the plant G is a proper and real-rational transfer matrix that is both stabilizable and detectable, it can be factored into a right coprime factorization (RCF) and a dual

left coprime factorization (LCF) [14, 127] .

$$G(z) = N(z)M^{-1}(z) \quad \text{RCF} \tag{5}$$

$$G(z) = \tilde{M}^{-1}(z)\tilde{N}(z) \quad \text{LCF}$$

These factorizations satisfy the Bezout Identity [14, 126] in Equation 6.

$$\begin{bmatrix} \tilde{X} & -\tilde{Y} \\ \tilde{M} & \tilde{N} \end{bmatrix} \begin{bmatrix} N & Y \\ -M & X \end{bmatrix} = I \tag{6}$$

From the factorization and the Bezout Identity, the set of all proper controllers which achieve internal stability are parameterized by the following [14, 323] :

$$\begin{aligned} K &= (-X + MZ_r)(Y + NZ_r)^{-1} \\ K &= \left(\tilde{Y} + Z_l\tilde{N}\right)^{-1}\left(-\tilde{X} + Z_l\tilde{M}\right) \end{aligned} \tag{7}$$

$$X, Y, \tilde{X}, \tilde{Y}, M, N, \tilde{M}, \tilde{N}, Z_r, Z_l \in RH_\infty.$$

Before using Equations 6 and 7 it is necessary to first define the discrete plant G and introduce a feedback variable v.

$$G \equiv \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \tag{8}$$

$$x(k+1) = Ax(k) + Bu(k) \tag{9}$$

$$y(k) = Cx(k) + Du(k) \tag{10}$$

Now introduce a state feedback variable, $v(k)$, according to

$$v(k) = Z_r^{-1}\left[u(k) - F_r x(k)\right] \text{ or} \tag{11}$$

$$u(k) = F_r x(k) + Z_r v(k). \tag{12}$$

Substituting Equation 12 back into 9 and 10 the new system in terms of the state feedback variable $v(k)$ is

$$x(k+1) = \left[A + BF_r\right] x(k) + BZ_r v(k) \tag{13}$$

$$y(k) = \left[C + DF_r\right] x(k) + DZ_r v(k) \tag{14}$$

Realizing this in terms of a RCF

$$\begin{bmatrix} M \\ N \end{bmatrix} \equiv \left[\begin{array}{c|c} A+BF_r & BZ_r \\ \hline F_r & Z_r \\ C+DF_r & DZ_r \end{array}\right] \tag{15}$$

yields the following results

$$\begin{aligned} u &= Mv \\ v &= M^{-1}u \\ y &= Nv \\ y &= NM^{-1}u. \end{aligned} \tag{16}$$

F_r is then chosen so that $A + BF_r$ has stable poles. In past research ([4] , [6]) these poles have been chosen to be at the origin, yielding Finite Impulse Response (FIR) behavior. Here, however, the only requirement is that the poles are stable, i.e. they must be within the unit circle.

Solving the dual problem

$$\begin{bmatrix} \tilde{M} & \tilde{N} \end{bmatrix} \equiv \left[\begin{array}{c|cc} A+LC & L & B+LD \\ \hline Z_lC & Z_l & Z_lD \end{array}\right] \tag{17}$$

where the poles of $A + LC$ are chosen so the system is stable.

2.2.3 State Prediction

Before the reference signal, v, can be input to the system from the optimizer, it is necessary first to calculate that signal from known information. Expanding upon Equations 12, 13, and 14, the discrete time state estimates are written as

$$\hat{x}(k+1) = A\hat{x}(k) + Bu(k) + L\left[\hat{y}(k) - y(k)\right] \tag{18}$$

but the estimated

$$\hat{y}(k) = C\hat{x}(k) + Du(k) \tag{19}$$

and

$$u(k) = F_r\hat{x}(k) + Z_rv(k)$$

yields

$$\hat{x}(k+1) = [A+BF_r+LC+LDF_r]\,\hat{x}(k)+[BZ_r+LDZ_r]\,v(k)-Ly(k). \quad (20)$$

Equation 20 can be rewritten for any time step in the future as

$$\hat{x}(k+l+1) = F(l)\hat{x}(k+l)+G(l)v(k+l)+H(l)y(k+l) \quad (21)$$

$$l = 0\ldots p-1$$

where

$$F(0) = A+BF_r+LC+LDF_r \quad (22)$$

$$F(m) = A+BF_r\ ,\ m=1\ldots l$$

$$G(l) = BZ_r+LDZ_r \quad (23)$$

$$H(0) = -L$$

$$H(m) = 0\ ,\ m=1\ldots l \quad (24)$$

The reason that $H(m)=0$ is because there are no future measured outputs, only predictions. Now it is possible to arrange equations 21, 22, 23 and 24 to produce a vector of future predicted states. At this point the assumption that D is zero for all physical systems is imposed. Rewriting Equation 21

$$\left\{\begin{array}{c}\bar{x}(k+1)\\ \vdots \\ \bar{x}(k+p)\end{array}\right\} = \tilde{F}\bar{x}(k)+\tilde{G}\bar{v}(k)+\tilde{G}^{\infty}\bar{v}^{\infty}(k)+\tilde{H}y(k) \quad (25)$$

where

$$\tilde{F} = \left[\begin{array}{c} F(0) \\ F(1)F(0) \\ \vdots \\ \prod_{j=p-1}^{0} F(j) \end{array}\right]$$

$$\tilde{G} = \left[\begin{array}{cccc} G & 0 & \cdots & 0 \\ F(1)G & G & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \left[\prod_{j=r}^{1} F(j)\right]G & \left[\prod_{j=r}^{2} F(j)\right]G & \vdots & G \\ \vdots & \vdots & \ddots & \vdots \\ \left[\prod_{j=p-1}^{1} F(j)\right]G & \left[\prod_{j=p-1}^{2} F(j)\right]G & \cdots & \left[\prod_{j=p-1}^{r} F(j)\right]G \end{array}\right]$$

$$\tilde{G}^{\infty} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ G & 0 & \cdots & 0 \\ F(r+1)G & G & \cdots & 0 \\ \left[\prod_{j=r+2}^{r+1} F(j)\right] G & F(r+2)G & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \left[\prod_{j=p-1}^{r+1} F(j)\right] G & \left[\prod_{j=p-1}^{r+2} F(j)\right] G & \cdots & \left[\prod_{j=p-1}^{\rho} F(j)\right] G \end{bmatrix} \quad (26)$$

$$\tilde{H} = \begin{bmatrix} H(0) \\ F(1)H(0) \\ \vdots \\ \left[\prod_{j=p-1}^{1} F(j)\right] H(0) \end{bmatrix}.$$

From Equation 12, any future control input is written as

$$u(k+l) = F_r \hat{x}(k+l) + Z_r v(k+l) \quad (27)$$

$$l = 0 \ldots q-1$$

$u(k+l)$ can now be written in vector form as

$$\begin{Bmatrix} u(k) \\ \vdots \\ u(k+q-1) \end{Bmatrix} = \tilde{F}_u \hat{x}(k) + \tilde{G}_u \bar{v}(k) + \tilde{G}_u^{\infty} \bar{v}^{\infty}(k) + \tilde{H}_u y(k) \quad (28)$$

where

$$\tilde{F}_u = \begin{bmatrix} F_r \\ F_r F(0) \\ \vdots \\ F_r \left[\prod_{j=q-2}^{0} F(j)\right] \end{bmatrix}$$

$$
\tilde{G}_u = \begin{bmatrix}
Z_r & 0 & \cdots & 0 \\
F_rG & Z_r & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
F_r\left[\prod_{j=r-2}^{1} F(j)\right]G & F_r\left[\prod_{j=r-2}^{2} F(j)\right]G & \cdots & Z_r \\
F_r\left[\prod_{j=r-1}^{1} F(j)\right]G & F_r\left[\prod_{j=r-1}^{2} F(j)\right]G & \cdots & F_rG \\
\vdots & \vdots & \ddots & \vdots \\
F_r\left[\prod_{j=q-2}^{1} F(j)\right]G & F_r\left[\prod_{j=q-2}^{2} F(j)\right]G & \cdots & F_r\left[\prod_{j=q-2}^{r} F(j)\right]G
\end{bmatrix}
$$

$$
\tilde{G}_u^{\infty} = \begin{bmatrix}
0 & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
Z_r & 0 & \cdots & 0 \\
F_rG & Z_r & \cdots & 0 \\
F_rF(r+1)G & F_rG & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
F_r\left[\prod_{j=q-2}^{r+1} F(j)\right]G & F_r\left[\prod_{j=q-2}^{r+2} F(j)\right]G & \cdots & F_r\left[\prod_{j=q-2}^{\rho} F(j)\right]G
\end{bmatrix} \tag{29}
$$

$$
\tilde{H}_u = \begin{bmatrix}
0 \\
F_rH(0) \\
\vdots \\
F_r\left[\prod_{j=q-2}^{1} F(j)\right]H(0)
\end{bmatrix}.
$$

From the cost function, Equation 3, a method for determining Δu is required. Letting

$$
\Delta u(k) = u(k) - u(k-1) \tag{30}
$$

the following relationships are formed.

$$
\begin{Bmatrix} \Delta u(k) \\ \vdots \\ \Delta u(k+q-1) \end{Bmatrix} = \tilde{F}_{\Delta}\hat{x}(k) + \tilde{G}_{\Delta}\tilde{v}(k) + \tilde{G}_{\Delta}^{\infty}\tilde{v}^{\infty}(k) + \tilde{H}_{\Delta}y(k) + \tilde{I}u(k-1) \tag{31}
$$

where

$$\tilde{F}_\Delta = \begin{bmatrix} F_r \\ F_rF(0) - F_r \\ F_rF(1)F(0) - F_rF(0) \\ \vdots \\ F_r\left[\prod_{j=q-3}^{0} F(j) - \prod_{j=q-4}^{0} F(j)\right] \\ F_r\left[\prod_{j=q-2}^{0} F(j) - \prod_{j=q-3}^{0} F(j)\right] \end{bmatrix}$$

$$\tilde{G}_\Delta = \begin{bmatrix} Z_r & \cdots & 0 \\ F_rG - Z_r & \cdots & 0 \\ F_rF(1)G - F_rG & \cdots & 0 \\ \vdots & \ddots & \vdots \\ F_r\left[\prod_{j=r-2}^{1} F(j) - \prod_{j=r-3}^{1} F(j)\right]G & \cdots & Z_r \\ F_r\left[\prod_{j=r-1}^{1} F(j) - \prod_{j=r-2}^{1} F(j)\right]G & \cdots & F_rG - Z_r \\ \vdots & \ddots & \vdots \\ F_r\left[\prod_{j=q-3}^{1} F(j) - \prod_{j=q-4}^{1} F(j)\right]G & \cdots & F_r\left[\prod_{j=q-3}^{r} F(j) - \prod_{j=q-4}^{r} F(j)\right]G \\ F_r\left[\prod_{j=q-2}^{1} F(j) - \prod_{j=q-3}^{1} F(j)\right]G & \cdots & F_r\left[\prod_{j=q-2}^{r} F(j) - \prod_{j=q-3}^{r} F(j)\right]G \end{bmatrix}$$

$$\tilde{G}_\Delta^\infty = \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ Z_r & \cdots & 0 \\ F_rG - Z_r & \cdots & 0 \\ F_rF(r+1)G - F_rG & \cdots & 0 \\ \vdots & \ddots & \vdots \\ F_r\left[\prod_{j=q-2}^{r+1} F(j) - \prod_{j=q-3}^{r+1} F(j)\right]G & \cdots & F_r\left[\prod_{j=q-2}^{\rho} F(j) - \prod_{j=q-3}^{\rho} F(j)\right]G \end{bmatrix}$$

$$\tilde{H}_\Delta = \begin{bmatrix} 0 \\ F_rH(0) \\ F_rF(0)H(0) - F_rH(0) \\ \vdots \\ F_r\left[\prod_{j=q-2}^{1} F(j) - \prod_{j=q-3}^{1} F(j)\right]H(0) \end{bmatrix} \tag{32}$$

$$\tilde{I} = \begin{bmatrix} -I \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Note, these prediction matrices are based on the absolute control input, u, being input into the plant. Previous works [4] and [6] , have modified the plant to accept incremental control inputs, Δu.

2.2.4 Quadratic Programing Problem

To solve the cost function of Equation 3, MATLAB's quadratic programing algorithm, qp.m, was used. The routine is based upon the following relationship

$$\begin{aligned} J(x) &= \min\left\{\frac{1}{2}x^T H x + f^T x\right\} \\ \text{subject to } Ax &\leq b \end{aligned} \tag{33}$$

The equations for the state predictions, incremental control inputs and system constraints must be rewritten to agree with the format of Equation 33. A new vector reference signal, $\tilde{v}(k)$, is defined in Equation 34 and is the vector over which the cost function is minimized.

$$\tilde{v}(k) = \begin{bmatrix} v(k)^T & v(k+1)^T & \cdots & v(k+r-1)^T \end{bmatrix}^T \tag{34}$$

2.2.4.1 Quadratic Programing Cost Function

Expanding the MIMO cost function of Equation 3 and using the state prediction matrices

$$J(k) = \tilde{v}(k)^T S \tilde{v}(k) + \begin{bmatrix} \hat{x}(k)^T & y(k)^T & \tilde{v}^{\infty}(k)^T & \tilde{s}(k)^T \end{bmatrix} T \tilde{v}(k) + ind \tag{35}$$

where S, T and ind are given by

$$S = \tilde{G}^T \tilde{C}^T \tilde{R}_y \tilde{C} \tilde{G} + \tilde{G}_{\Delta}^T \tilde{R}_u \tilde{G}_{\Delta} \tag{36}$$

$$T = 2 \begin{bmatrix} \tilde{C}\tilde{F} \\ \tilde{C}\tilde{H} \\ \tilde{C}\tilde{G}^{\infty} \\ -I_{p\eta} \end{bmatrix} \tilde{R}_y \tilde{C} \tilde{G} + 2 \begin{bmatrix} \tilde{F}_{\Delta} \\ \tilde{H}_{\Delta} \\ \tilde{G}_{\Delta}^{\infty} \\ 0 \end{bmatrix} \tilde{R}_u \tilde{G}_{\Delta} \tag{37}$$

$$\begin{aligned} ind = {} & \left(\tilde{F}\hat{x}(k) + \tilde{G}^{\infty}\tilde{v}^{\infty} + \tilde{H}y(k)\right)^T \tilde{C}^T \tilde{R}_y \tilde{C} \left(\tilde{F}\hat{x}(k) + \tilde{G}^{\infty}\tilde{v}^{\infty} + \tilde{H}y(k)\right) + \\ & \left(\tilde{F}_{\Delta}\hat{x}(k) + \tilde{G}_{\Delta}^{\infty}\tilde{v}^{\infty} + \tilde{H}_{\Delta}y(k)\right)^T \tilde{C}^T \tilde{R}_u \tilde{C} \left(\tilde{F}_{\Delta}\hat{x}(k) + \tilde{G}_{\Delta}^{\infty}\tilde{v}^{\infty} + \tilde{H}_{\Delta}y(k)\right) + \\ & \tilde{s}(k)^T \tilde{R}_y \left[\tilde{s}(k) - 2\tilde{C}\left(\tilde{F}\hat{x}(k) + \tilde{H}y(k) + \tilde{G}^{\infty}\tilde{v}^{\infty}\right)\right] \end{aligned} \tag{38}$$

Since ind is independent of the optimization variable $\tilde{v}$, it was neglected. Additionally, $\tilde{C}$, $\tilde{R}_y$, and $\tilde{R}_u$ are defined as

$$\tilde{C} = \text{diag}\,(C,\ldots,C) \tag{39}$$

$$\tilde{R}_y = \text{diag}\,(R_y,\ldots,R_y) \tag{40}$$

$$\tilde{R}_u = \text{diag}\,(R_u,\ldots,R_u) \tag{41}$$

The far future reference input, v^{∞}, is calculated using Lemma 5.1 in [6] .

$$C\sum_{j=1}^{(\rho-r)}(A+BF_r)^{j-1}BZ_rv^{\infty} = s^{\infty} \tag{42}$$

The purpose of calculating v^{∞} is discussed in the Pole Placement for System Stability and Steady State Error section.

2.2.4.2 Quadratic Programing Constraints

Ideally there would be unlimited control power and the system would be able to handle unlimited outputs. However, all physical systems have limits on their control inputs and occasionally they have implied limits on the system outputs. For aerospace systems control inputs are limited to their maximum and minimum position limits as well as maximum and minimum rates. Various outputs may also be limited based upon potential damage to the system or exceeding human factor constraints. It is the ability of MPC strategy to handle system constraints within the control problem which makes it appealing for highly dynamic systems. System constraints are expressed across the control and prediction horizons as

$$\begin{aligned}
\Delta u_{min}(k+i) &\leq \Delta u(k+i) \leq \Delta u_{max}(k+i) \\
i &= 0\ldots q-1 \\
u_{min}(k+i) &\leq u(k+i) \leq u_{max}(k+i) \\
i &= 0\ldots q-1 \\
\hat{x}_{min}(k+i) &\leq \hat{x}(k+i) \leq \hat{x}_{max}(k+i)
\end{aligned} \tag{43}$$

$$i = 1 \ldots p$$

where q and p are the control and prediction horizons. System outputs can be determined by a linear combination of states using Equation 10, thereby allowing output constraints as well as state constraints.

Defining the system constraints to match MATLAB's quadratic programing format in Equation 33, the following relationships are constructed [4]

$$\begin{aligned} L_\lambda \Delta\tilde{u}(k) &\leq l(k) \\ M_\mu \tilde{u}(k) &\leq m(k) \\ N_\nu \begin{bmatrix} \hat{x}(k+1) \\ \vdots \\ \hat{x}(k+p) \end{bmatrix} &\leq n(k) \end{aligned} \tag{44}$$

where

$$\begin{aligned} L_\lambda &= \begin{bmatrix} -I_{\xi,q} \\ I_{\xi,q} \end{bmatrix} \\ M_\mu &= \begin{bmatrix} -I_{\xi,q} \\ I_{\xi,q} \end{bmatrix} \\ N_\nu &= \begin{bmatrix} -I_{\kappa,p} \\ I_{\kappa,p} \end{bmatrix} \end{aligned} \tag{45}$$

and

$$\begin{aligned} l(k) &= \begin{bmatrix} -\Delta u_{min} \\ \Delta u_{max} \end{bmatrix} \\ m(k) &= \begin{bmatrix} -u_{min} \\ u_{max} \end{bmatrix} \\ n(k) &= \begin{bmatrix} -\hat{x}_{min} \\ \hat{x}_{max} \end{bmatrix}. \end{aligned} \tag{46}$$

Using the prediction matrices of Equations 26, 29, and 32 the following relationships are defined.

$$\begin{aligned} \tilde{D}\tilde{v}(k) &\leq \tilde{E}c(k) \\ \tilde{D} &= \begin{bmatrix} L_\lambda \tilde{G}_\Delta \\ M_\mu \tilde{G}_u \\ N_\nu \tilde{G} \end{bmatrix} \\ \tilde{E} &= \begin{bmatrix} -L_\lambda \tilde{F}_\Delta & -L_\lambda \tilde{G}_\Delta^\infty & -L_\lambda \tilde{H}_\Delta & -L_\lambda & I & 0 & 0 \\ -M_\mu \tilde{F}_u & -M_\mu \tilde{G}_u^\infty & -M_\mu \tilde{H}_u & 0 & 0 & I & 0 \\ -N_\nu \tilde{F} & -N_\nu \tilde{G}^\infty & -N_\nu \tilde{H} & 0 & 0 & 0 & I \end{bmatrix} \end{aligned} \tag{47}$$

$$c(k) \equiv \left[\hat{x}(k)^T \quad v^{\infty}(k)^T \quad y(k)^T \quad u(k-1)^T \quad l(k)^T \quad m(k)^T \quad n(k)^T \right]^T$$

2.3 Pole Placement for System Stability and Steady State Error

Recent work [4] and [6] has focused on placing the poles of observer and state estimator controllers ,Equations 15 and 17, at the origin for discrete systems. This has assured mathematically that the system would reach a desired setpoint in a finite amount of time. However, this assumes that all states are modeled and the plant is an LTI model. Using real systems, not all the states are modeled, noise sources introduce error and generally systems are nonlinear and time varying. Because of these potential problems, controller pole placement was relaxed from the placement of discrete poles at the origin (continuous at $-\infty$) to within the unit circle (stable). Because of this relaxation, it was not possible to guarantee trajectory tracking of a LTI model within a finite amount of time. However it was mathematically possible to reach a window around the tracking point in a finite amount of time.

From reference [6] , Lemma 5.1, a relationship between the far future reference input, v^{∞}, and the far future setpoint trajectory, s^{∞}, is given by

$$C\sum_{j=1}^{2\kappa}(A+BF_r)^{j-1}BZ_r v^{\infty} = s^{\infty}. \tag{48}$$

This is the result that was used in Equation 42 to solve for v^{∞}. Equation 48 is based upon the poles of $A+BF_r$ and $A+LC$ being zero. However in this research the discrete poles were simply made stable. Therefore it was necessary to construct a relationship between controller pole placement, prediction/control horizons, and setpoint error. Starting with Equation 5.36 of reference [6] (shown in Equation 49), it is possible to determine the error of the stable system.

$$\begin{aligned}\begin{bmatrix} \hat{x}(k+l) \\ \tilde{x}(k+l) \end{bmatrix} &= \begin{bmatrix} A+BF_r & -LC \\ 0 & A+LC \end{bmatrix}^{l} \begin{bmatrix} \hat{x}(k) \\ \tilde{x}(k) \end{bmatrix} + \\ &\sum_{j=1}^{l} \begin{bmatrix} A+BF_r & -LC \\ 0 & A+LC \end{bmatrix}^{j-1} \begin{bmatrix} BZ_r \\ 0 \end{bmatrix} v(k+l-j) \end{aligned} \tag{49}$$

$\hat{x}(k+l) \equiv$ Future Estimated States

$$\tilde{x}(k+l) \equiv x(k+l) - \hat{x}(k+l) \tag{50}$$

For a system where the controller poles are at the origin and letting $l \geq 2\kappa$ [6] yields

$$\begin{bmatrix} A+BF_r & -LC \\ 0 & A+LC \end{bmatrix}^{2\kappa} = 0. \tag{51}$$

Knowing the far future setpoint trajectory, s^{∞}, the far future reference input is calculated using Equation 48. For a simply stable system

$$\begin{bmatrix} A+BF_r & -LC \\ 0 & A+LC \end{bmatrix}^{l} \rightarrow 0 \quad \text{as} \quad l \rightarrow \infty. \tag{52}$$

Since it is impractical to let $l \rightarrow \infty$, a heuristic approach to determining the error using a first order SISO system was derived and then applied to the MIMO system.

First recall the following definitions.

$$\begin{aligned} p &\equiv \text{prediction horizon} \\ q &\equiv \text{control horizon} \\ r &\equiv \text{optimization horizon} \\ T_s &\equiv \text{Discrete Time Step} \\ \rho &\equiv \max(p,q) \end{aligned} \tag{53}$$

Also let the dynamic setpoint trajectory be fixed to s^{∞} at the end of the optimization horizon, r. Letting

$$l = \rho - r \tag{54}$$

provides l time steps for the error to reach a finite value. Let the error be defined as

$$e_{rr} \equiv 1 - w \tag{55}$$

where

$$w \equiv \text{User defined Window of Steady State Error.} \tag{56}$$

For a first order continuous system of the form

$$H(s) = \frac{p_c}{s + p_c} \tag{57}$$

subject to a step input the error is defined as

$$e_{rr} = e^{p_c t}. \tag{58}$$

Setting the time $t = l * T_s$ and solving for the continuous pole

$$p_c = \frac{\ln(1-w)}{lT_s}. \tag{59}$$

Now recalling the relationship between continuous and discrete pole locations

$$p_c = \frac{\ln(z)}{T_s} \tag{60}$$

and combining with Equation 59, a relationship between pole placement, prediction/control horizons, and setpoint error is found

$$\begin{aligned} \frac{\ln(1-w)}{lT_s} &= \frac{\ln(z)}{T_s} \\ z &= e^{\frac{\ln(1-w)}{l}} \\ z^l &= (1-w) \end{aligned} \tag{61}$$

which is analogous to the MIMO Equation 52. As an example $z = 0.82$ given $w = 98\%$ and $l = 20$. For the MIMO system Equation 61 was used as a rough guide to place the poles of $A + BF_r$ and $A + LC$ such that

$$F_{r_i} \equiv \text{Initial Pole Location or Pole closest to Unit Circle for } A + BF_r$$

$$F_{r_i} = (1 - w_{F_r})^{\frac{1}{l}} \tag{62}$$

$$L_i \equiv \text{Initial Pole Location or Pole closest to Unit Circle for } A + LC$$

$$L_i = (1 - w_L)^{\frac{1}{l}}. \tag{63}$$

All other controller poles were then placed closer to the origin than their respective starting locations. The placement of these poles was accomplished using Ackerman's Formula.

2.4 Tuning Parameters

Having established the mathematical development of the MPC strategy it is beneficial to list the various tuning parameters in Table 1.

Table 1. Tuning Parameters

Tuning Parameter	**Description**
R_u	Control Input Weighting Matrix
R_y	Tracking Error Weighting Matrix
$A + BF_r$ Pole locations	State Gains of Inner Feedback Loop
$A + LC$ Pole locations	State Estimator of Inner Feedback Loop
T_s	Discrete Time Step
p, q, r	Prediction, Control and Optimation Horizons
l, m, n	Control Rate and Limit Constraints and System Constraints
s	Commanded Trajectory

Currently specific relationships between system tracking and required control power are not developed in terms of these tuning parameters. Relationships are generally developed using a heuristic approach and numerous simulations.

Chapter 3 - Plant Models, Trajectories and MPC State Space Formulation

3.1 F-16 Model

3.1.1 Linear Model

Generation of linear models was performed using various MATLAB function calls and script files listed in Appendix A. Several of these script files and function calls were based upon the FORTRAN code developed by Dr. B.L. Stevens [12] and transcribed into MATLAB format by Dr. Brad Liebst and several of his students [8] . A new front end script file, Modgen.m in Appendix A, was written to automate the generation of linear models. Modgen.m requires a user defined input matrix given in the script file case1.m, whose rows correspond to different flight conditions at which linear models are to be generated. The four elements in each row correspond to Mach number, bank angle in degrees, rate of climb in ft/sec and altitude in ft. Modgen.m then calls the MATLAB function trimmer.m [3] , which calculates the control inputs necessary to make the state derivatives zero. Next, Modgen.m calls jacob.m [2] , which calculates the linearized state space model based upon the trimmed flight condition from trimmer.m. Finally Modgen.m outputs the state space matrices and equilibrium values into a file unique to the specified flight condition. Both trimmer.m and jacob.m are designed to use any nonlinear aircraft model obeying the equation

$$\dot{x} = f(x, u). \tag{64}$$

3.1.2 F-16 Nonlinear Model

The nonlinear MATLAB function subf16a.m [13] , provides the $\dot{x}$ vector specified in Equation 64. The code utilizes extensive look up tables based upon wind-tunnel data developed by NASA-Langley[12, 124] . The nonlinear model currently has time invariant physical properties which are

listed in Table 2. The actuator physical limits and rates are listed in Table 3. The nonlinear model, subf16a.m, has limitations which are important to remember when performing simulations. These limitations, listed in Table 4, are based upon the bounds of the wind tunnel data which was collected by NASA-Langley. Control input rate and position constraints, vectors l and m, are taken from the information in Table 3.

Table 2. F-16 Physical Properties

Property	**Value**
Weight	$20,500\ lbs$
Moment of Inertia, J_{xx}	$9,496\ Slug * ft^2$
Moment of Inertia, J_{yy}	$55,814\ Slug * ft^2$
Moment of Inertia, J_{zz}	$63,100\ Slug * ft^2$
Moment of Inertia, J_{xz}	$982\ Slug * ft^2$
Span	$30.0\ ft$
Area	$300\ ft^2$
Mean Aerodyanmic Center	$11.32\ ft$

Table 3. Control Actuator Data

Actuator	Deflection Limit	Rate Limit	Time Constant
Elevator	$\pm 25.0°$	$60°/sec$	0.0495 sec lag
Ailerons	$\pm 21.5°$	$80°/sec$	0.0495 sec lag
Rudder	$\pm 30.0°$	$120°/sec$	0.0495 sec lag
Throttle No Afterburner	$0 \ldots 0.7699$	$100\frac{1}{sec}$	0.0 sec lag
Throttle, With Afterburner	$0.77 \ldots 1.0$	$100\frac{1}{sec}$	0.0 sec lag

Table 4. Nonlinear F-16 Model Limitations

Parameter	**Minimum Limit**	**Maximum Limit**
Throttle Position	0.0	1.0
Altitude, h	$0\ ft$	$50,000\ ft$
Mach Number	0.0	$\simeq 0.6$
Angle of Attack, α	$-10°$	$45°$
Angle of Side Slip, β	$-30°$	$-30°$

The throttle rate limit, Δu_T, was set to an arbitrary large value, $100\frac{1}{sec}$. This was done to simulate a near instantaneous throttle input response. However, the engine dynamics are modeled

with an appropriate delay, so that there is not an instantaneous change in thrust. Note that the throttle input, u_T, is unitless as it represents a percentage of engine thrust.

3.1.3 Plant Block Diagrams

In order to simplify the controller block diagram and assist in the modular design, the plant model was grouped into a single SIMULINK subsystem. A specific plant, linear or nonlinear, with the correct number of inputs and outputs could easily be placed into the complete MPC controller.

3.1.3.1 F-16 Linear Block Diagram

Figure 1 shows the linear model configuration. It is a standard state space configuration except that in the SIMULINK Discrete Plant block $C = I$ and $D = 0$. D was set to zero because physical systems tend to have an inherent delay in any input to output. Also the generation of the linear models by trimmer.m calculates $D = 0$ for states which do not involve acceleration. Setting $C = I$ provides the designer the flexibility of calculating the outputs through a linear combination of the states, x, by using SIMULINK matrix gain blocks, K. These outputs are then multiplexed in the order in which the MPC controller is designed.

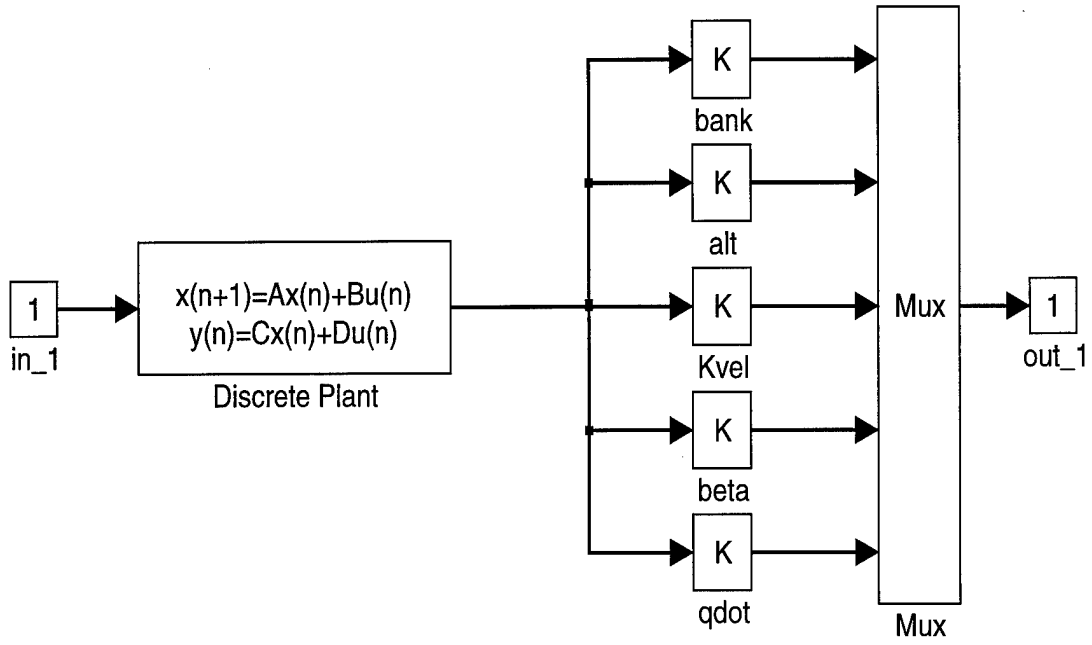

Figure 1. F-16 Linear Model Block Diagram

3.1.3.2 F-16 Nonlinear Block Diagram

Figure 2 shows the SIMULINK block diagram used for the nonlinear model.

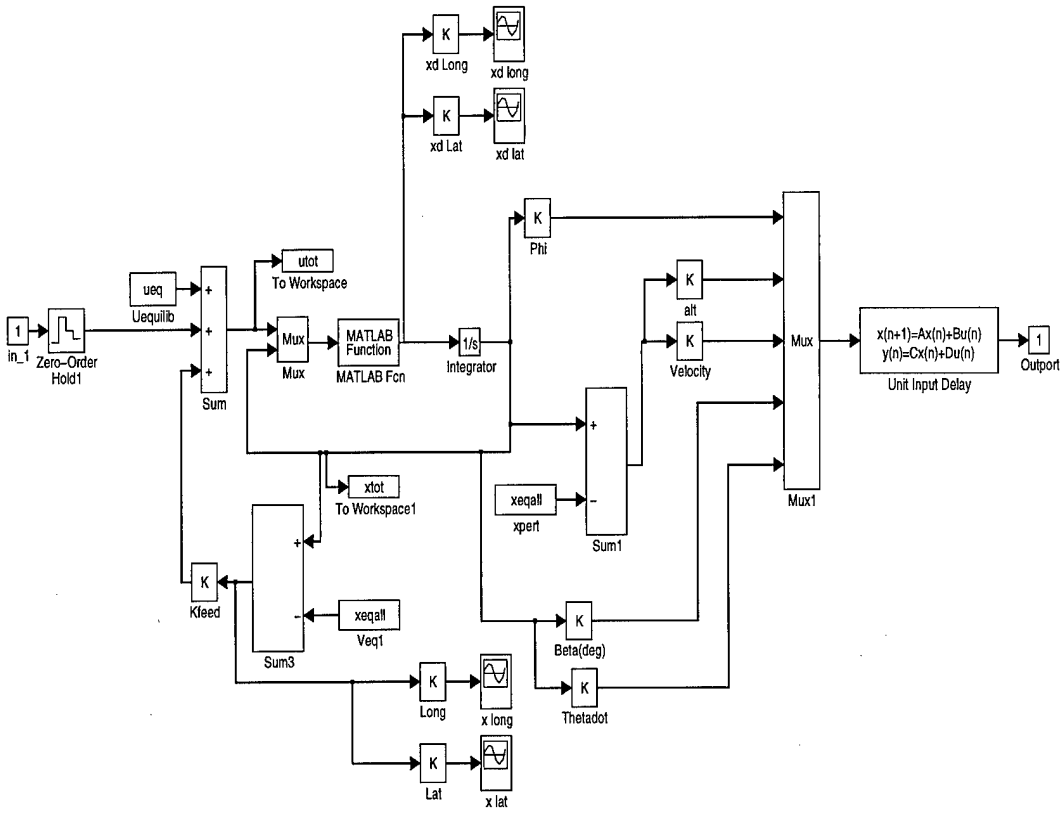

Figure 2. Nonlinear F-16 Model Block Diagram

The input to this block diagram is the output of the controller, u, which is fed into a zero order hold. The zero order hold ensures that the continuous integration performed by the nonlinear plant is fed a discrete input for each time step, T_s. Since u is a perturbation signal and subf16a.m operates on total inputs, the equilibrium input value, u_0, is added to u before being input into the MATLAB function block. The heart of this diagram is the MATLAB function block where the nonlinear state derivatives are calculated using subf16a.m. The state derivatives are then integrated, in the integrator block $\frac{1}{s}$, using a MATLAB proprietary 5^{th} order Runge Kutta equation solver. The total states are then fed back into the MATLAB function block. The initial conditions, in the integrator block, are

set to the equilibrium state values, $x_{eq_{all}}$, calculated by trimmer.m. The four graphical output blocks *xd long*, *xd lat*, *x long* and *x lat* are used to display the lateral and longitudinal states during simulations. The specific lateral and longitudinal states are selected using the matrix gain blocks, K, preceding these graphical blocks. Note that the longitudinal state outputs are perturbation outputs since $x_{eq_{all}}$ was subtracted in the Sum3 block. Figure 2 also includes a state feedback block, *Kfeed*. This block was included in the design to allow for continuous stabilizing feedback but was set to zero in this thesis. As in the linear block diagram, Figure 1, the specific outputs were calculated by a linear combination of the states determined by the SIMULINK matrix gain blocks *Phi*, *alt*, *Velocity*, *Beta* and *Thetadot*. These outputs were then multiplexed to provide the correct output vector, y, to the MPC controller. Finally the unit delay block preceding the output provides the controller with the current output vector while a new vector is being calculated by the continuous integration.

3.1.4 Initial Conditions

The initial conditions, Table 5, for all the simulations were chosen to simulate realistic flight conditions. A clean F-16 has an optimum climb airspeed of $\approx 400\ knots$. Since the nonlinear model is limited to a Mach number of ≈ 0.6, a low starting altitude was chosen to achieve an airspeed close to $400\ knots$.

Table 5. Initial Flight Conditions

Parameter	**Value**
Mach	0.6
Altitude, h	$100\ ft$
Velocity	$390.6\ knots\ 669.7\ ft/sec$
Bank Angle, ϕ	$0°$
Rate of Climb, $\dot{h}$	$0\ ft/sec$
C.G. Location	$30\% mac$

The control input equilibrium values are given in Equation 65 for a 30% c.g. location.

$$u_0 = \begin{bmatrix} 0.262 & -1.544° & 0° & 0° \end{bmatrix}^T \tag{65}$$

where

$$u \equiv \begin{bmatrix} u_T & u_{el} & u_{ail} & u_{rud} \end{bmatrix}^T$$

Using Modgen.m, linear models were generated using the initial conditions in Table 5 and varying $\dot{h}$. The corresponding eigenvalues are given in Table 6.

Table 6. Eigenvalues at Mach = 0.6, Alt = 100ft, Bank Angle = 0 deg, C.G. = 0.30

Longitudinal*				
Rate of Climb	$\dot{h} = 0ft/sec$	$\dot{h} = 100ft/sec$	$\dot{h} = 200ft/sec$	$\dot{h} = 300ft/sec$
Short Period	$-1.59 \pm 1.99i$	$-1.59 \pm 1.99i$	$-1.59 \pm 1.99i$	$-1.59 \pm 1.99i$
Phugoid	$-0.013 \pm 0.055i$	$-0.0091 \pm 0.053i$	$-0.0056 \pm 0.051i$	$-0.0021 \pm 0.047i$
Engine	-1.0	-1.0	-1.0	-5.0
Lateral*				
Dutch Roll	$-0.54 \pm 4.12i$	$-0.54 \pm 4.12i$	$-0.55 \pm 4.11i$	$-0.55 \pm 4.10i$
Roll	-4.96	-4.96	-4.97	-4.98
Spiral	-0.0104	-0.0033	$+0.0039$	$+0.012$
* Eigenvalues are in continuous time with units rad/sec				

Note that the eigenvalues vary slightly as $\dot{h}$ varies. However at $\dot{h} = 300ft/sec$, there is a sudden change in the engine dynamics. This represents a transition from military power to afterburner. This discrete jump occurs in the nonlinear model, subf16a.m, when the throttle input is 0.77 or greater. To avoid problems associated with this discrete jump two additional constraints were placed upon the simulations. First, the throttle was limited to $u_{T_{max}} = 0.765$. Second, the altitude trajectories were designed for $\dot{h} \leq 290\ ft/sec$ based upon the results of Table 7.

Table 7. Throttle Position for Different $\dot{h}$ Values

$C.G. = 30\%$	$\dot{h} = 290ft/sec$	$u_{throtle} = 0.764$
	$\dot{h} = 300ft/sec$	$u_{throtle} = 0.774$

3.1.5 Actuator Constraints

As discussed, the throttle actuator was limited to $u_{T_{max}} = 0.765$. The rudder was also limited due to the initial flight condition. In actual flight, the rudder is seldom used beyond a degree or

two at flight conditions other than landing configurations. Because this is an up and away flight condition (i.e. no flaps, no gear, much greater than $1g$ stall velocity), the rudder was limited to $\pm 0.5°$. Table 8 shows the actual constraints which were used in simulations.

Table 8. MPC Actuator Constraints

Actuator	Deflection Limit	Rate Limit	Time Constant
Elevator	$\pm 25.0°$	$60°/sec$	0.0495 sec lag
Ailerons	$\pm 21.5°$	$80°/sec$	0.0495 sec lag
Rudder	$\pm 0.5°$	$120°/sec$	0.0495 sec lag
Throttle No Afterburner	$0 \ldots 0.765$	$100\frac{1}{sec}$	0.0 sec lag

Since the controller is designed to operate on perturbation inputs, it is necessary to redefine the constraints in terms of the physical limits and equilibrium values.

$$\begin{aligned} m_{max} &= u_{max} - u_0 \\ m_{min} &= u_{min} - u_0 \end{aligned} \tag{66}$$

Since the equilibrium condition implies $\dot{u} = 0$, there was no modification of the controller input rate limits, l_{max} or l_{min}.

3.2 Dynamic Trajectories

The MPC controller setup that was used in this thesis assumes that there was a future control input, v^{∞}, which was found by a constant future setpoint trajectory s^{∞}. In past research [4] and [6] , this future setpoint trajectory was simply the step input that was commanded. However, for a dynamic trajectory a method of determining s^{∞} is required. The method chosen was to let $s^{\infty} \equiv constant$ after the optimization horizon, r, has passed.

$$\begin{aligned} s^{\infty}(l) &= s(k+r) \\ r+1 &\leq l \leq \rho \end{aligned} \tag{67}$$

Figure 3 displays a graphical based upon

$$r = 10$$

$$p = 30 \tag{68}$$

$$q = 30.$$

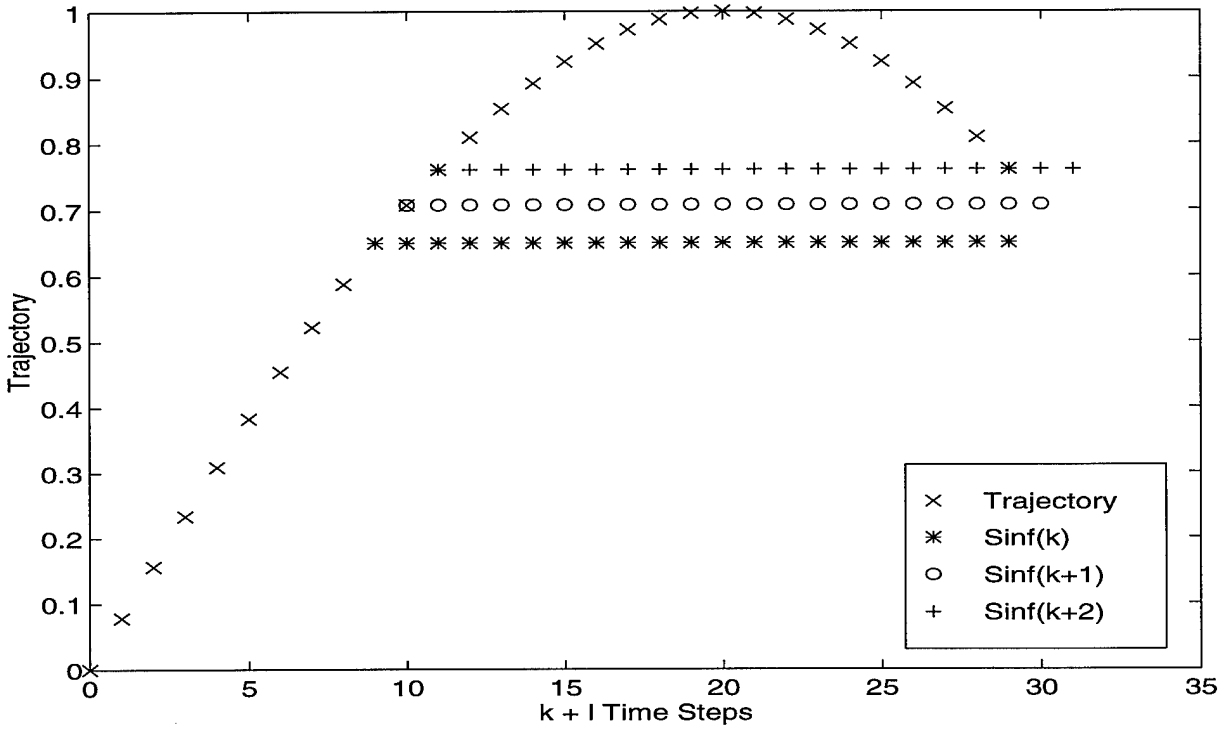

Figure 3. Optimization and Far Future Trajectory

The dynamic trajectories are generated at each time step using the MATLAB function file trajpnt.m. Trajpnt.m is called during the optimization at each time step. It is envisioned that instead of a prescribed trajectory which is known at the start of the simulation, pilot modeled inputs could be substituted for trajpnt.m.

3.2.1 Bank Angle

Two different trajectories were used for the desired dynamic bank angle trajectories. The first was a simple half sine wave, shown in Figure 4, according to

$$s_\phi(t) = \begin{cases} \phi_{max}\left(\sin\left(\frac{\pi}{t_{ss}}t\right)\right), & t < t_{ss} \\ 0, & t \geq t_{ss} \end{cases} \tag{69}$$

$$\phi_{max} \equiv \text{Maximum Desired Bank Angle.}$$

Since roll rate, $\dot{\phi}$, is strongly coupled with aileron input [9, Chapter 5] and it is desirable to generate smooth control inputs, a second trajectory where the starting and ending derivatives are zero was produced using a cosine wave, as shown in Equation 70 and Figure 5:

$$s_\phi(t) = \begin{cases} \frac{\phi_{max}}{2}\left(1 - \cos\left(\frac{2\pi}{t_{ss}}t\right)\right), & t < t_{ss} \\ 0, & t \geq t_{ss} \end{cases} \tag{70}$$

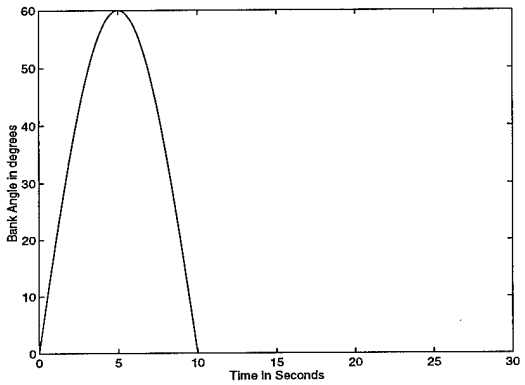

Figure 4. Bank Angle Trajectory - Half Sine Wave, tss = 10sec

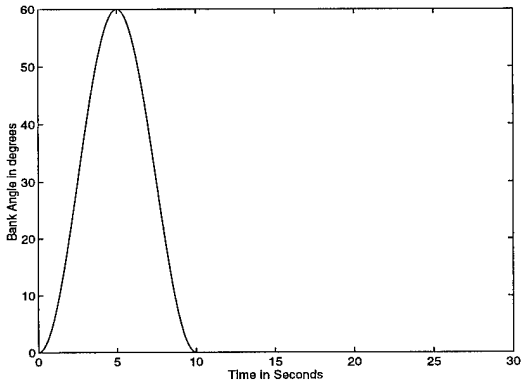

Figure 5. Bank Angle Trajectory - Full Cosine Wave, tss = 10sec

3.2.2 Altitude

For altitude transitions a smooth trajectory was also desired. Modifying Equation 70, the altitude equation is given by and shown in Figure 6:

$$s_{alt}(t) = \begin{cases} \frac{h_{max}}{2}\left(1 - cos\left(\frac{\pi}{t_{ss}}t\right)\right), & t < t_{ss} \\ h_{max}, & t \geq t_{ss} \end{cases} \tag{71}$$

$$h_{max} \equiv \text{Maximum Desired Altitude.}$$

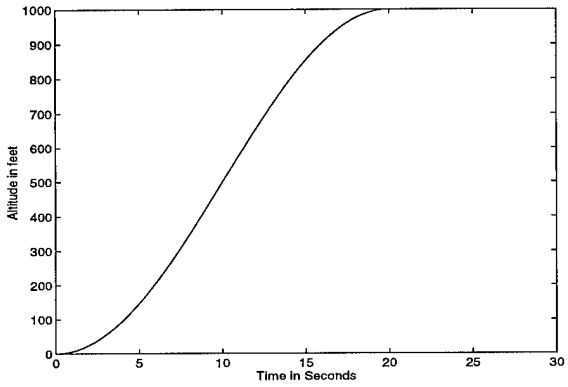

Figure 6. Altitude Trajectory - Half Cosine Wave, tss = 20sec

3.3 MPC State Space Formulation

The following three sections describe in detail the block diagrams, controller pole placement technique and the adaptive constraint techniques utilized in the simulations.

3.3.1 Block Diagrams

The entire controller incorporating the plant is shown in the SIMULINK block diagram of Figure 7:

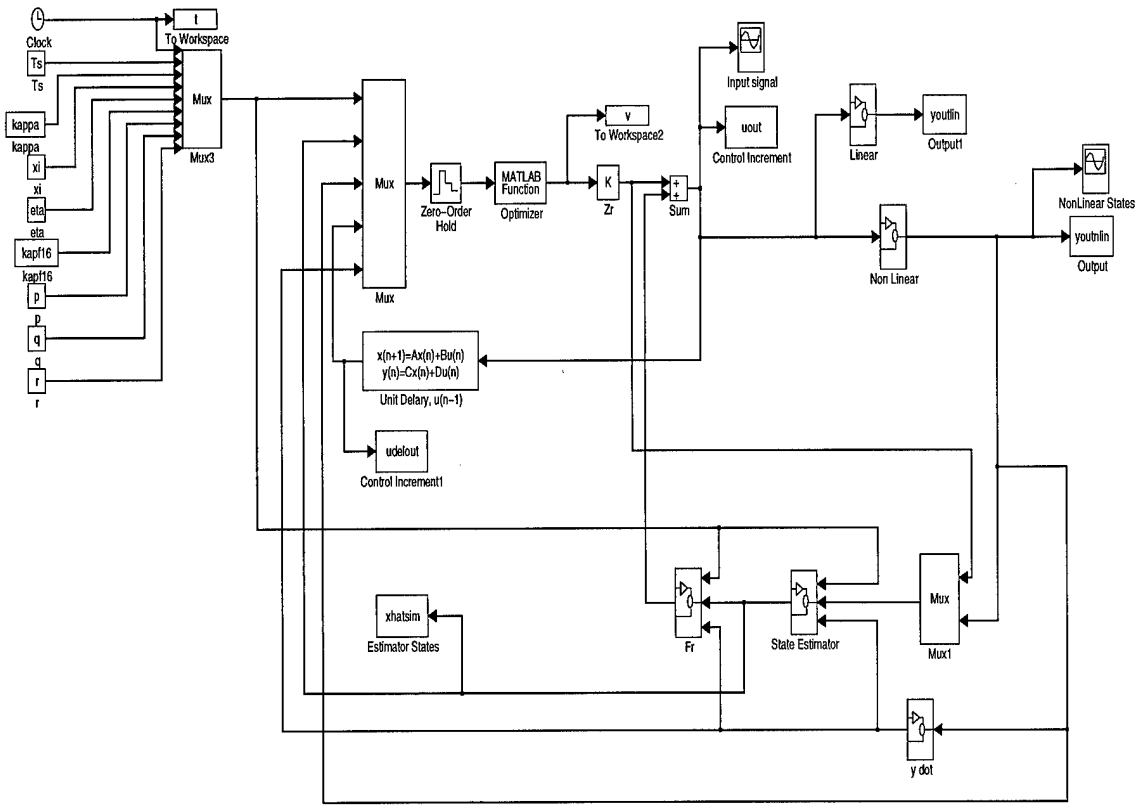

Figure 7. Model Predictive Control - Overall Simulink Block Diagram

The block diagram is divided into five main areas. The first is the actual plant. Shown in the loop is the SIMULINK icon *Non Linear*, which contains the nonlinear F-16 model. As discussed, any linear or nonlinear plant can be substituted there provided the individual elements of the plant

output vector correspond to the desired trajectory. As configured, the plant output vector is

$$y(k) = \begin{bmatrix} \phi & h & V & \beta & q \end{bmatrix}^T \tag{72}$$

where units used are degrees, feet and seconds. The output vector in Equation 72 can be changed. If it is changed, the corresponding 'desired trajectory' vector, s, must also be changed accordingly in the MATLAB code trajpnt.m.

Continuing around the loop from the plant, the state estimator is reached. The estimator design is based upon the results of Chapter 2. The estimator operates on the plant outputs, y, and the plant inputs, u. Its state space representation is given by

$$G \equiv \left[\begin{array}{c|cc} A+BF+LC & B & -L \\ \hline I & 0 & 0 \end{array}\right]. \tag{73}$$

Note the gains for $A+BF_r$ and $A+LC$ were chosen based upon a linearized model at the operating point described in Table 5. Figure 8 shows the block diagram used to realize Equation 73. The three SIMULINK input blocks *in_1*, *in_2* and *in_3* correspond to a vector of constants, the control input vector $\begin{bmatrix} u & y \end{bmatrix}$ and a user definable gain scheduling parameter. The two MATLAB function blocks, *A Matrix* and *B Matrix*, call the MATLAB functions Amatrix.m and Bmatrix.m. These function calls are used to select the LTI state space matrices A, B, C, F_r and L. The output of the state estimator is the vector $\hat{x}$.

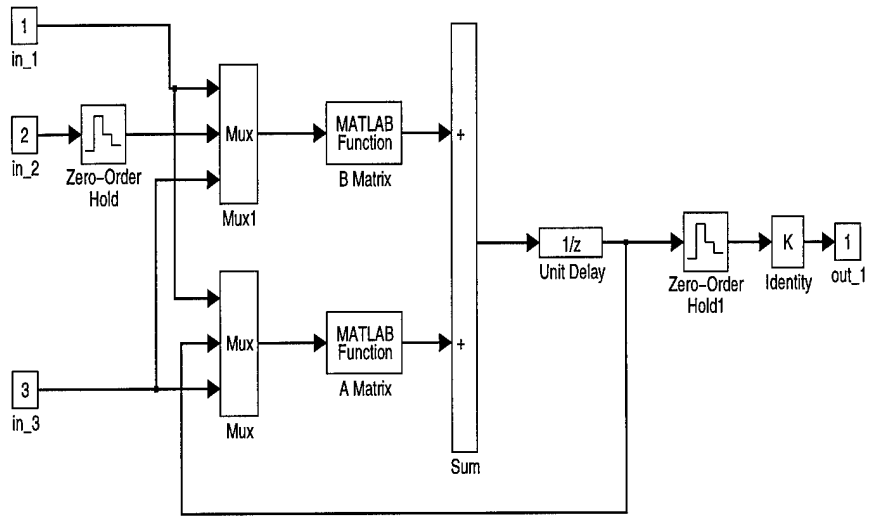

Figure 8. Simulink State Estimator Block Diagram

The controller was designed with the flexibility to perform on-line gain scheduling based upon a user defined input parameter. Examples of these parameters could include y, $\dot{y}$, u or others. Whenever a linearized state space matrix or prediction matrix is required by any portion of the MPC controller, a MATLAB function call is made to matget.m. Currently matget.m returns the LTI state space matrices and prediction matrices which were calculated based upon the initial conditions of Table 5.

The next point in the loop is the gain block, F_r, for the estimated states, $\hat{x}$. The MATLAB function block in Figure 9 calls the routine Frwhich.m, which determines the gain F_r through matget.m. Then the estimated states, $\hat{x}$, are multiplied by F_r and are output to the summing block preceding the plant input. The three input blocks, *in_1*, *in_2* and *in_3*, correspond to a vector of constants, the state estimates, $\hat{x}$, and a gain scheduling parameter.

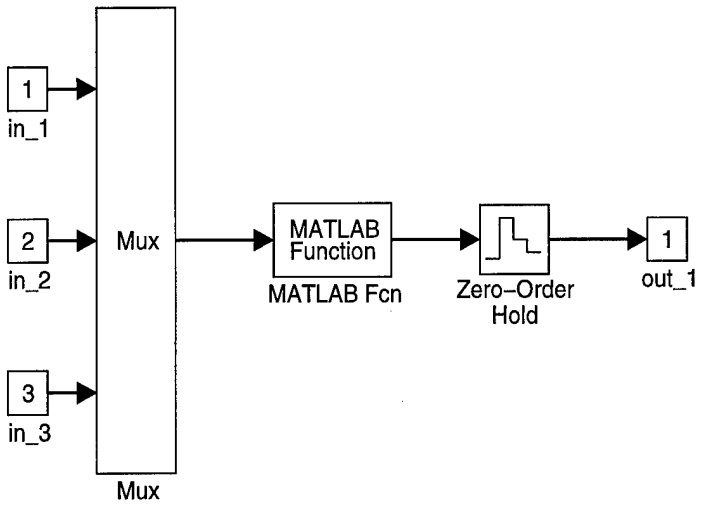

Figure 9. State Feedback Gains - Simulink Block Diagram

The outputs of the plant, state estimator, the previous control input, $u\,(k-1)$, and user defined control parameter are then fed into the *Optimizer* block. The zero order hold is placed at the input of the *optimizer* to ensure that the *optimizer* works on a discrete basis. The output of the *optimizer* is the reference signal v. This is fed into the summing block and a perturbation control signal is fed into the plant. The code, optimize.m in Appendix A, calls matget.m to determine the LTI state space and prediction matrices. Also, adaptive constraint techniques are employed within optimize.m, if selected.

The fifth section of Figure 7, located at the upper left hand corner, is comprised of the various constants used throughout the MPC controller. These constants are multiplexed into a vector and then distributed to the various MPC controller sections.

3.3.2 Pole Placement Technique

As presented in Chapter 2, controller pole placement was not restricted to the origin in this thesis. Using Ackerman's Formula [5, page 497] a pole placement routine was written, gains3.m, listed in Appendix A. The routine requires starting locations, F_{r_i} and L_i, for poles closest to the unit circle of $A + BF_{r_i}$ and $A + LC$ and the discrete time step, T_s. The routine then generates a vector of pole locations in the continuous domain where the bounds are given by Equation 74. The vector is the length of the number of controller states, κ.

$$\begin{aligned} |P_{min}| &= \left| \frac{\ln(F_{r_i})}{T_s} \right| \\ |P_{max}| &= \left| \frac{\ln(PerF_{r_i})}{T_s} \right| \\ Per &\equiv \text{User Defined Percentage} \end{aligned} \tag{74}$$

Once the gain matrices, F_r and L, have been generated, the discrete pole locations are recomputed. If the pole closet to the unit circle is greater than F_{r_i} or L_i, the corresponding starting value is multiplied by the percentage R and the appropriate gain matrix is recalculated. For this thesis Per and R equalled 95% and 98%.

3.3.3 Control Input Constraint Weighting Matrices

In Chapter 2 the control input constraint weighting matrices were defined as shown in Equation 75.

$$\begin{aligned} L_\lambda &= \begin{bmatrix} -I_{\xi,q} \\ I_{\xi,q} \end{bmatrix} \\ M_\mu &= \begin{bmatrix} -I_{\xi,q} \\ I_{\xi,q} \end{bmatrix} \\ N_\nu &= \begin{bmatrix} -I_{\kappa,p} \\ I_{\kappa,p} \end{bmatrix} \end{aligned} \tag{75}$$

However, for the simulations Equation 76 lists the actual values assigned to these weighting matrices.

$$L_\lambda = \begin{bmatrix} -I_{\xi,\xi} & 0_{\xi,(q-1)} \\ I_{\xi,\xi} & 0_{\xi,(q-1)} \end{bmatrix}$$

$$
\begin{aligned}
M_\mu &= \begin{bmatrix} -I_{\xi,\xi} & 0_{\xi,(q-1)} \\ I_{\xi,\xi} & 0_{\xi,(q-1)} \end{bmatrix} \\
N_\nu &= [\,], \text{Null Matrix}
\end{aligned} \tag{76}
$$

By choosing N_ν to be a null matrix system states are unconstrained. The structure of L_λ and M_μ places constraints on the control inputs at the time step, k, but during the optimization problem future control inputs of $k+l$ for $l = 1 \ldots q$ are unconstrained. The choice of these constraint matrices was designed to provide an aggressive control scheme for a highly maneuverable aircraft.

3.3.4 Adaptive Constraint Techniques

Two adaptive constraint techniques were derived to assist the optimization routine in finding the global minimum at each time step. Initial simulations involving the LTI model showed a relationship between roll rate, $\dot{\phi}$, and aileron input. This agreed with conventional thinking [9, 75-77] and for a simplified model the relationship between roll rate, $\dot{\phi}$, and perturbation aileron input, $\Delta\delta_a$, is given by [9, 154, Equation 5.6]

$$
\dot{\phi} = \frac{-L_{\delta_a}}{L_p}\Delta\delta_a \tag{77}
$$

Using a linear simulation, bank angle was commanded to follow a sine wave input as described in Equation 69. The resulting bank angle output, $\phi(t)$, was differentiated and plotted against aileron control inputs. A linear fit was applied to the data and Equation 78 was the result.

$$
\begin{aligned}
\dot{\phi} &= -16.7\tfrac{deg}{sec*deg}\Delta\delta_a, \quad u_a \equiv \Delta\delta_a \\
u_a &= -0.06\dot{\phi}
\end{aligned} \tag{78}
$$

In the code, optimize.m, $\dot{\phi}$ was approximated by a first order finite difference method. Also an estimated error bank rate and an approximated rate of climb were all calculated as given in Equation 79.

$$
\begin{aligned}
\dot{\phi}_p &\equiv \frac{s_\phi(k+1) - s_\phi(k)}{T_s} \\
\dot{\phi}_e &\equiv \frac{s_\phi(k+1) - y_\phi(k)}{T_s}
\end{aligned} \tag{79}
$$

$$\dot{h}_p \equiv \frac{s_h(k+1) - s_h(k)}{T_s}$$

The adaptive aileron constraints were then calculated based upon three separate conditions listed in Table 9.

Table 9. Adaptive Aileron Selection Criteria

Case	$\dot{h}_p$	$\dot{\phi}_p$	Equation Number
1	0	0	80
2	0	$\neq 0$	81
3	$\neq 0$	$\neq 0$	82

Based upon the conditions of the three cases, new maximum and minimum actuator constraint limits are defined. In Equations 80 through 82, the prime, m'_{max_a} and m'_{min_a}, notation indicates that a new value for the constraint has been assigned. The maximum and minimum values m_{max_a} and m_{min_a}, correspond to the original actuator maximum and minimum constraint values.

$$\begin{aligned} m'_{max_a} &= \frac{-0.06}{2}\dot{\phi}_e + \frac{0.25}{100}m_{max_a} \\ m'_{min_a} &= \frac{-0.06}{2}\dot{\phi}_e + \frac{0.25}{100}m_{min_a} \end{aligned} \tag{80}$$

$$\begin{aligned} m'_{max_a} &= -0.06\dot{\phi}_p + \frac{0.5}{100}m_{max_a} \\ m'_{min_a} &= -0.06\dot{\phi}_p + \frac{0.5}{100}m_{min_a} \end{aligned} \tag{81}$$

$$\begin{aligned} m'_{max_a} &= -0.06\dot{\phi}_p + \frac{-0.06}{2}\dot{\phi}_e + \frac{0.5}{100}m_{max_a} \\ m'_{min_a} &= -0.06\dot{\phi}_p + \frac{-0.06}{2}\dot{\phi}_e + \frac{0.5}{100}m_{min_a} \end{aligned} \tag{82}$$

The 0.5% buffer was added to prevent over constraining the optimization problem.

The elevator adaptive constraint technique was derived in a similar manner. The altitude output, h, was commanded to follow a cosine trajectory as defined in Equation 71. The second derivative of $h(t)$ was then taken and plotted against the elevator input signal and a linear fit of the data yielded

the following

$$u_{el} = -\frac{\ddot{s}_{alt}}{45}. \tag{83}$$

A 2% to 5% buffer was then added to this heuristic equation to derive the following adaptive elevator constraints

$$\begin{aligned} m'_{max_{el}} &= -\frac{\ddot{h}}{45} + \frac{2.0}{100} m_{max_{el}} \\ m'_{min_{el}} &= -\frac{\ddot{h}}{45} - \frac{2.0}{100} m_{min_{el}}. \end{aligned} \tag{84}$$

The buffer of 2% to 5% was selected for two specific reasons. First there was no consideration of throttle input in deriving Equation 83. This introduces error into the assumption that $\ddot{h}$ is dependent solely on elevator deflection since classical performance techniques have shown that $\dot{h}$ is heavily dependent upon excess thrust, thus implying that throttle inputs are important. The second reason for the buffer was because there was no consideration of required elevator during bank angle changes. One potential fix to the neglected bank angle affect is to derive a linear relationship between bank angle and elevator input and then add that relationship to Equation 83.

3.4 Operating Envelope and F-16 Views

In Appendix C are figures of the F-16 operating envelope and the planform and longitudinal views of the F-16 [1] . The planform and longitudinal views of the F-16 were included merely as a reference for the reader. The operating envelope shows the specific excess power, P_s, curves for a clean F-16 at military power. This chart was included to show that the initial conditions provide a point which is well within the operating envelope of the F-16. This is important as the trajectories designed for this thesis attempted to simulate realistic conditions showing that the MPC strategy is viable for an aerospace system. If the initial conditions were on the edge of the operating envelope, the realism of the simulations would be in question. Note that for a steady level turn of 60° (2 g's) at a Mach number of 0.6, P_s is almost 300 ft/sec. This implies that for constant altitude bank

angle trajectories where the bank angle does not exceed 60°, the throttle should never be required to exceed the 0.77 setting corresponding to military power.

Chapter 4 - Results of Simulations

Before presenting the results of the simulations it is important to understand three basic concepts. First, the MPC controller consisting of the MIMO cost function, state prediction matrices and system constraints used for all simulations, was designed using an LTI F-16 model at the initial conditions presented in Chapter 3. Results which are described as *linear* in nature imply that the plant of the entire system was the same LTI F-16 model used for the construction of the MPC controller. Simulation results which are described as *nonlinear*, imply that the plant of the system is the nonlinear F-16 model shown in Chapter 3 using the MATLAB code subf16.m.

4.1 Initial Validation

Before attempting to use a new controller on a new aircraft model, a validation of the MPC controller design and dynamic trajectory tracking was required. As much of the ground work necessary for this thesis was completed by reference [4] , the LTI model used in that thesis of the High Angle Research Vehicle (HARV) was used to validate the MPC controller. For a complete description of the HARV model, operating point, scaled inputs and scaled outputs, the reader is referred to reference [4] . The state space discrete realization is generated by a MATLAB script file, harv.m. The state space matrices generated are shown in Appendix C. Equation 85 lists the order of states, inputs and outputs with input descriptions identified in Table 10.

$$\begin{aligned} x_{long} &= \left[\, V_T \;\; \alpha \;\; q \;\; \theta \;\; h \,\right]^T \\ u_{long} &= \left[\, \delta_{TVS} \;\; \delta_{AS} \;\; \delta_{SS} \;\; \delta_{LES} \;\; \delta_{TES} \;\; \delta_T \,\right]^T \\ y &= \left[\, V_T \;\; \gamma \;\; \theta \,\right]^T \end{aligned} \tag{85}$$

Table 10. Validation Control Input Definitions

Control Input	Description
δ_{TVS}	Symmetric Thrust Vectoring
δ_{AS}	Symmetric Aileron
δ_{SS}	Symmetric Stabilator
δ_{LES}	Symmetric Leading Edge Flap
δ_{TES}	Symmetric Trailing Edge Flap
δ_T	Throttle

Equation 86 describes the trajectory commanded for the MPC controller validation.

$$s = \begin{bmatrix} \begin{cases} 0.6\sin\left(\frac{2\pi}{t_{ss}}t\right), & t_{ss} \leq 10sec \\ 0 & t_{ss} > 10sec \end{cases} \\ \begin{cases} 1.0\left(1-\cos\left(\frac{\pi}{t_{ss}}t\right)\right), & t_{ss} \leq 20sec \\ 0 & t_{ss} > 20sec \end{cases} \\ 0 \end{bmatrix}^T \tag{86}$$

As seen in Figure 10 the simulation yielded the expected sine and cosine shapes of Equation 86. Figures 11 and 12 show the corresponding control inputs. From Reference [4] , the position constraint limits of the thrust vectoring, δ_{TVS}, were set to ± 1 unit. Figure 11 shows that δ_{TVS} is an active constraint. Figure 12 also shows an active rate constraint on Leading Edge flaps, δ_{LES}, since its rate limit was set to $\pm 0.5\frac{units}{sec}$ in reference [4] . These active constraints serve to validate the correct functioning of the MPC controller design.

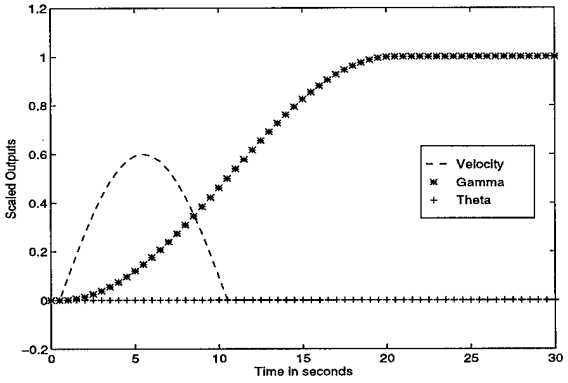

Figure 10. Longitudinal State Outputs of LTI HARV Model using Dynamic Trajectories

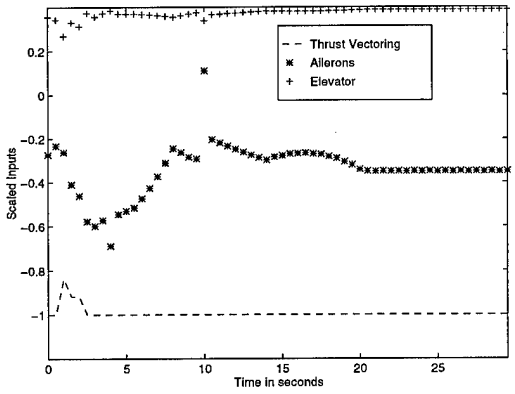

Figure 11. Control Inputs (1-3) of LTI HARV Model using Dynamic Trajectories

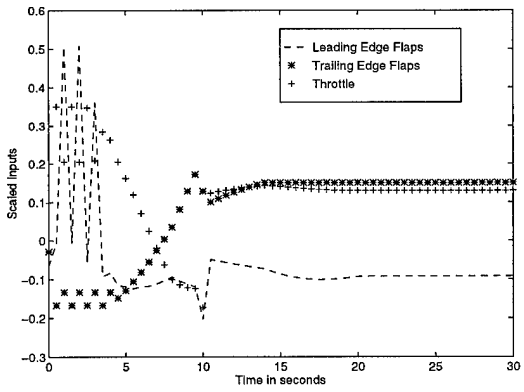

Figure 12. Control Inputs (4-6) of LTI HARV Model using Dynamic Trajectories

4.2 Linear Model Results

Using the discrete pole placement technique described in Chapter 3, a sample of the pole locations using $F_{r_i} = 0.75$ is given in Equation 87. Descriptions of simulations list the starting pole locations F_{r_i} and L_i.

$$eig(A+BF_r) = \begin{bmatrix} 0.750 & 0.746 & 0.742 & 0.737 & 0.733 & 0.729 & 0.725 & \dots \\ 0.721 & 0.717 & 0.713 \end{bmatrix}^T \quad (87)$$

Equation 88 provides the discrete time step, weighting matrices and output vector description which were used for all linear simulations. The aircraft model trim conditions were given in Chapter 3, Table 5, along with the control input equilibrium values in Equation 65.

$$\begin{aligned} T_s &= 0.1\ sec \\ R_u &= \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix} \\ R_y &= \begin{bmatrix} 100 & 0 & 0 \\ 0 & 1000 & 0 \\ 0 & 0 & 10000 \end{bmatrix} \\ y(k) &= \begin{bmatrix} \phi & h & V \end{bmatrix}^T \end{aligned} \quad (88)$$

4.2.1 Bank Angle Response

Seven separate cases, listed in Table 11, were simulated using the linear model to characterize the system response to different bank angle commanded trajectories. Cases 1 and 2 served to validate the assumption presented in Chapter 3 that rudder deflection constraints may be severely limited without sacrificing tracking, provided a landing task is not being simulated. Cases 2 and 3 further validate that simply stable pole placement provides adequate tracking. Cases 4 through 7 were used to identify the optimum bank angle trajectory shape. Also the advantages and disadvantages of the adaptive aileron constraint technique developed in Chapter 3 were identified. No linear simulations are presented here using the adaptive elevator constraint technique as it did not affect any of the linear results.

Table 11. Linear Model, Bank Angle Cases

Case	A*	B*	C*	D*
1	Off	0.75	$\pm 30°$	$60°$ Step Input
2	Off	0.75	$\pm 0.5°$	$60°$ Step Input
3	Off	0.50	$\pm 0.5°$	$60°$ Step Input
4	Off	0.75	$\pm 0.5°$	$60°$ Sin Input, $t_{ss} = 10sec$
5	On	0.75	$\pm 0.5°$	$60°$ Sin Input, $t_{ss} = 10sec$
6	Off	0.75	$\pm 0.5°$	$60°$ Cos Input, $t_{ss} = 10sec$
7	On	0.75	$\pm 0.5°$	$60°$ Cos Input, $t_{ss} = 10sec$
*A - Aileron Adaptive Constraint On or Off				
*B - $A + BF_r$ and $A + LC$ Pole Starting Location				
*C - Rudder Stops				
*D - Trajectory				

Comparing Figures 13 and 15, there is no noticeable difference in rise time or overshoot for a $60°$ step input despite the difference in rudder limit constraints. While the variation in steady state perturbation outputs V and h in Figures 14 and 16 are negligible compared to their respective trim conditions, the use of more restrictive rudder constraints did increase damping, overshoot and settling time, thus improving longitudinal output tracking. A comparison of Figure 15 and the Case 3 corresponding output (see Appendix D) showed that the more restrictive pole placement of Case

3 did not noticeably improve bank angle response. It did, however, slightly improve velocity and altitude response, but the steady state tracking response in Figure 16 was more than acceptable with

$$\Delta V_{ss} = -0.0005\%$$

$$\Delta h_{ss} = -0.01\%.$$

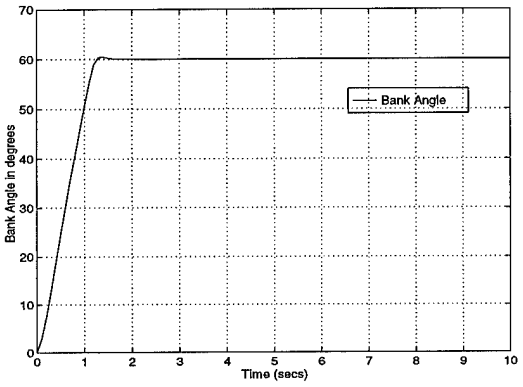

Figure 13. Case 1 - Bank Angle Output

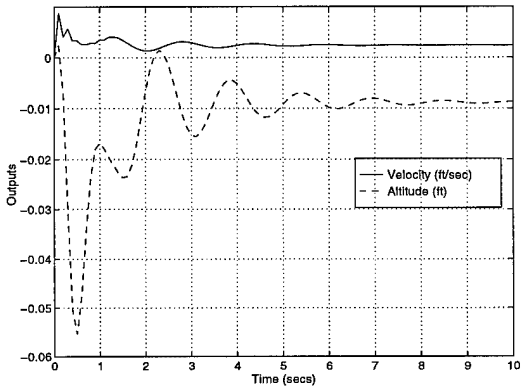

Figure 14. Case 1 - Velocity and Altitude Outputs

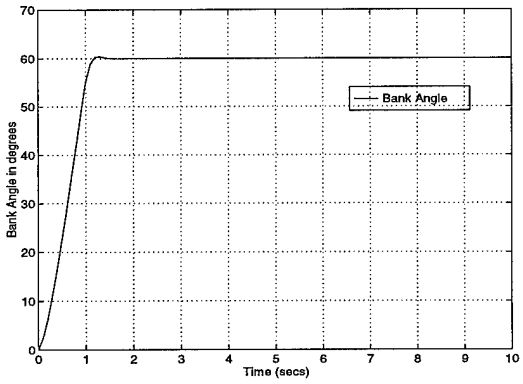

Figure 15. Case 2 - Bank Angle Output

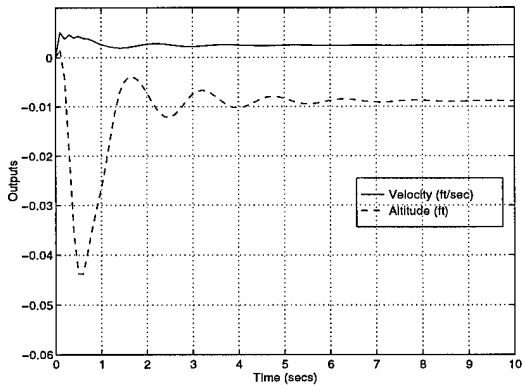

Figure 16. Case 2 - Velocity and Altitude Outputs

The use of more restrictive rudder constraints also showed positive benefits for the control inputs. Figure 20 shows there was less longitudinal control power required for the more restrictive rudder constraints as compared to Figure 17. There is a much more dramatic decrease in required lateral control power as seen in Figure 20. As seen before in the validation simulation results, Figure

20 shows the aileron rate constraint is active. Additionally Figure 20 shows the rudder position constraint is also active. The aileron is experiencing rate constraints according to

$$\begin{aligned} \Delta u_{max} &= \dot{u}_{max} * T_s \\ \Delta u_{min} &= \dot{u}_{min} * T_s \end{aligned} \tag{89}$$

Where $\dot{u}_{max_{ail}} = 80°/sec$ from Table 8 of Chapter 3 and $T_s = 0.1\ sec$ yields the maximum variation in aileron control input, Δu, between time steps is 8°. The ratcheting of the aileron control inputs is due to the unconstrained future control input $u(k+l)$, where $l = 1 \ldots q$, as discussed in Chapter 3. This ratcheting is not considered to be a problem as step inputs are not realistic trajectories and the results are presented here to demonstrate the capabilities of the MPC controller. If this ratcheting were to create problems one could redefine the constraint matrices, L_λ, M_μ, and N_ν to allow constraints at future time $k+l$. A limitation of the LTI model is shown in Figures 17 and 20. One can see the steady state values of elevator and throttle are both zero. An actual aircraft would require an increase in both (positive deflection of throttle and negative for elevator) to maintain constant altitude. As stated in the Introduction, this research concentrated on realistic outputs. As will be seen in the nonlinear simulations, the expected increase in throttle and elevator are achieved. The placement of the MPC controller pole locations closer to the origin was simulated in Case 3. The results, in Appendix D, did not provide any significant benefits over Case 2.

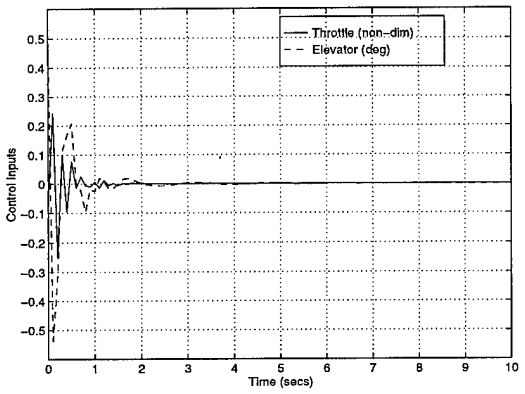

Figure 17. Case 1 - Longitudinal Controls

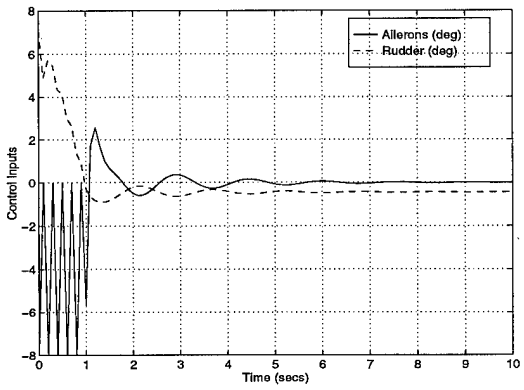

Figure 18. Case 1 - Lateral Controls

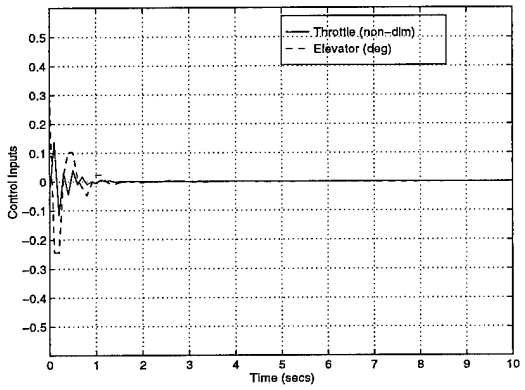

Figure 19. Case 2 - Longitudinal Controls

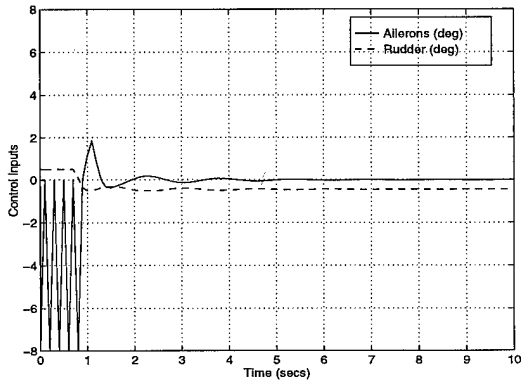

Figure 20. Case 2 - Lateral Controls

In Cases 4 and 5 a comparison was made between the use of absolute aileron constraints, Case 4, and the adaptive aileron constraint technique developed in Chapter 3, Case 5. The results are presented in Appendix D due to the unrealistic boundary conditions. The sine wave bank angle trajectory imposed an instantaneous jump in roll rate, which is unrealistic in an actual aircraft. The simulations did provide informative results as to the ability of the adaptive aileron constraint technique to handle these discrete boundary conditions. A comparison of Cases 4 and 5 showed that the adaptive constraint technique reduced smooth system tracking. This is attributed to the suspected reduction of stability caused by the adaptive aileron constraint technique. Also the required control power was increased. Despite these consequences of the adaptive aileron constraint technique,

simulation results are still acceptable. As will be shown later, as more demanding trajectories are commanded, the benefits of the adaptive constraint techniques outweigh these consequences.

In Cases 6 and 7, the bank angle was commanded to follow a cosine trajectory. The use of a cosine trajectory imposed zero roll rate boundary conditions, which provided a realistic bank angle trajectory. Due to the cosine trajectory, there was little difference between the results of Cases 6 and 7 shown in Figures 21 through 28. Figure 23 shows a much improved bank angle overshoot as compared to Case 5. Despite slightly increased control power usage due to the adaptive aileron technique, the results of Case 7 are more than acceptable.

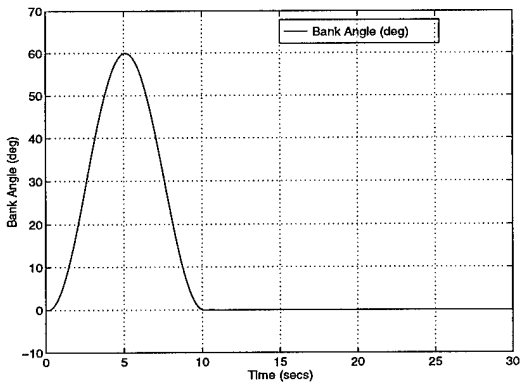

Figure 21. Case 6 - Bank Angle Output

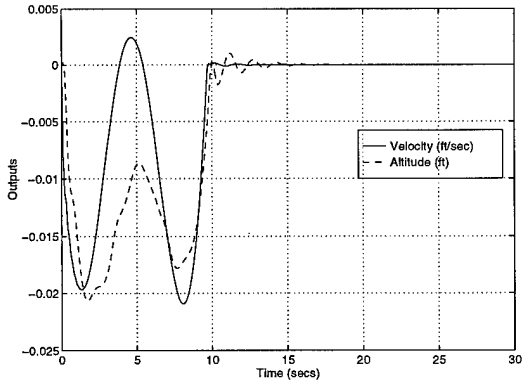

Figure 22. Case 6 - Velocity and Altitude Outputs

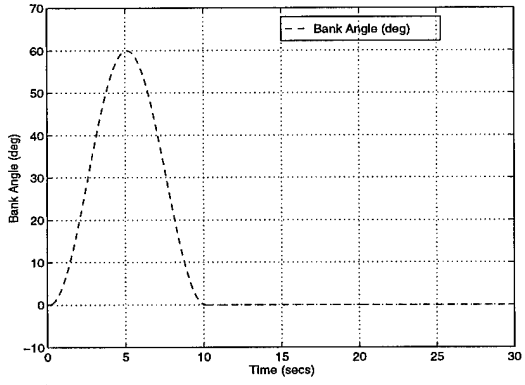

Figure 23. Case 7 - Bank Angle Output

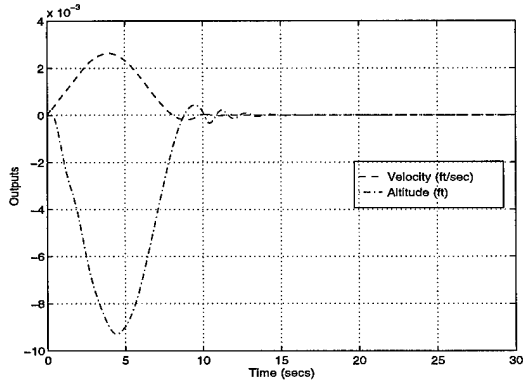

Figure 24. Case 7 - Velocity and Altitude Outputs

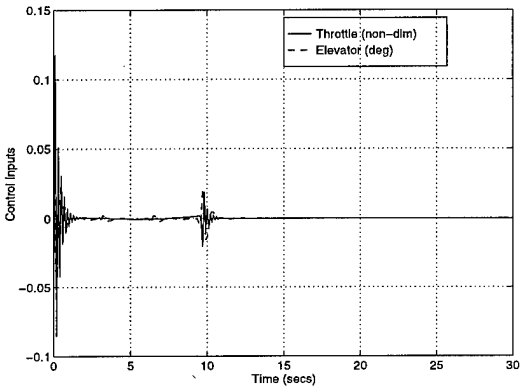

Figure 25. Case 6 - Longitudinal Controls

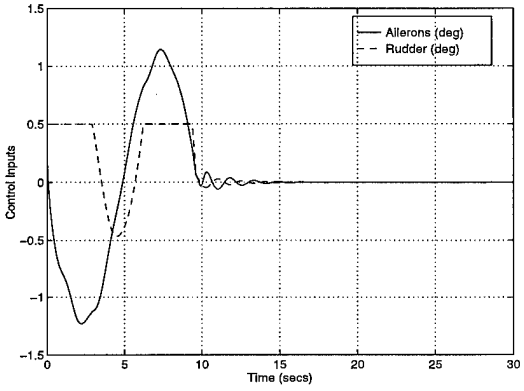

Figure 26. Case 6 - Lateral Controls

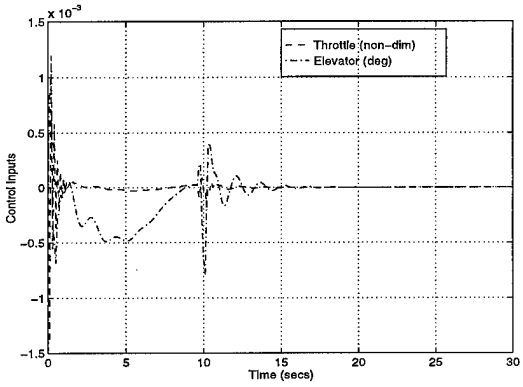

Figure 27. Case 7 - Longitudinal Controls

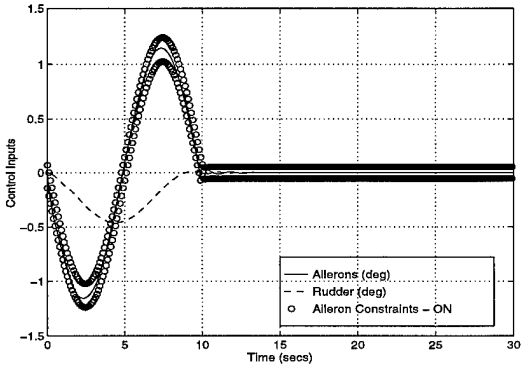

Figure 28. Case 7 - Lateral Controls

When the adaptive aileron constraint technique is employed, it is the $\dot{\phi}_e$ term, defined in Equation 79 of Chapter 3, which causes the oscillatory behavior. As a stability analysis was not performed on this term, these oscillations are expected. Additionally, the relative contribution of $\dot{\phi}_e$ to the overall adaptive technique can cause the system to go unstable. For this thesis a trial and error approach was used to develop the appropriate $\dot{\phi}_e$ contribution, trading decreased stability for increased tracking performance. Future research using adaptive constraints should address these stability issues directly.

4.2.2 Altitude Response

Three cases listed in Table 12 were simulated to characterize commanded altitude response. Since the constrained rudder, $\pm 0.5°$, showed positive results in the bank angle responses and rudder is primarily a lateral control input, it was once again constrained to $\pm 0.5°$. Additionally there was little noticeable improvement in placing the poles of the controller closer to the origin so their placement remained at $F_{r_i} = 0.75$ and $L_i = 0.75$.

Table 12. Linear Model, Altitude Cases

Case	**A***	**B***	**C***	**D***
8	Off	0.75	$\pm 0.5°$	$500ft$ Step Input
9	Off	0.75	$\pm 0.5°$	$500ft$ Cos Input, $t_{ss} = 20sec$
10	On	0.75	$\pm 0.5°$	$500ft$ Cos Input, $t_{ss} = 20sec$
*A - Aileron Adative Constraint On or Off				
*B - $A + BF_r$ and $A + LC$ Pole Starting Location				
*C - Rudder Stops				
*D - Trajectory				

Figures 29 through 32 show that the altitude response is at least initially unstable when commanded to a step trajectory. The simulation response is unrealistic as it demands an infinite rate of climb at the first time step. Since the control inputs were constrained, this implies that there may not be a feasible solution to the commanded step trajectory.

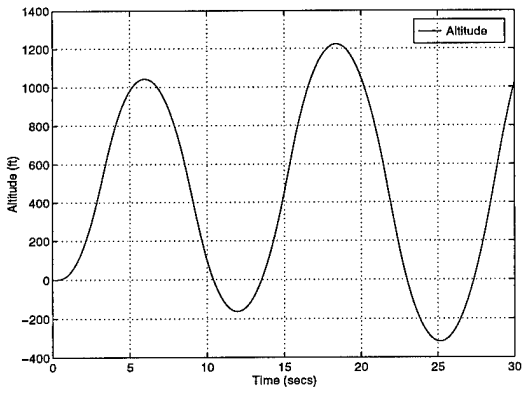

Figure 29. Case 8 - Altitude Output

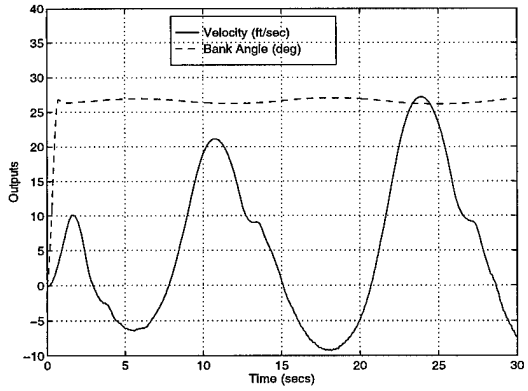

Figure 30. Case 8 - Velocity and Bank Angle Outputs

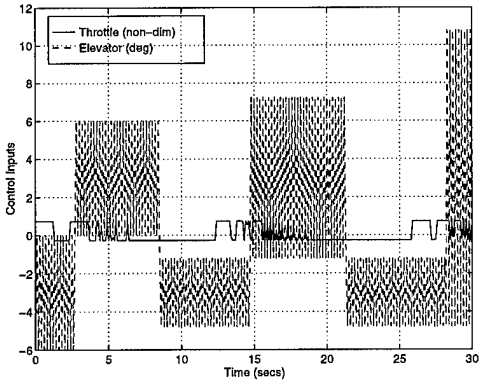

Figure 31. Case 8 - Longitudinal Controls

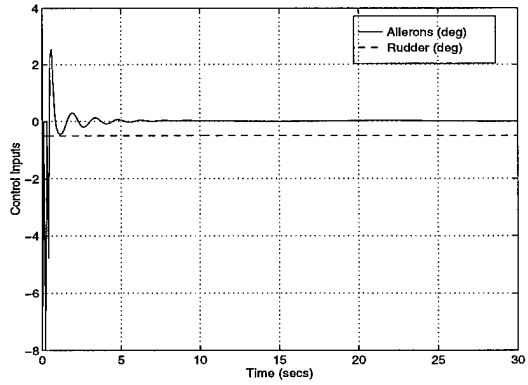

Figure 32. Case 8 - Lateral Controls

Given sufficient time, the system should in theory stabilize and converge since the linear theory states that the cost function is monotonically decreasing, however the response shown in Figure 29 would be objectionable to any pilot. The violent elevator oscillations would also be objectionable as they would decrease the life expectancy of the elevator actuators. The ratcheting of the elevator input can also be attributed to the unconstrained future control inputs $u(k+l)$ as discussed earlier. Because of these concerns a realistic cosine trajectory was commanded in Cases 9 and 10 according to Equation 71 of Chapter 3.

Figures 33 through 36 show the simulation results using a commanded cosine trajectory and the affect of using absolute aileron constraints versus adaptive constraints. In both cases altitude steady

state tracking error is less than 4%. However the big improvement of using adaptive constraints was the significantly reduced steady state bank angle error, 27° in Figure 34 and 0.5° in Figure 36. The corresponding control inputs for Cases 9 and 10 are shown in Figures 37 through 40.

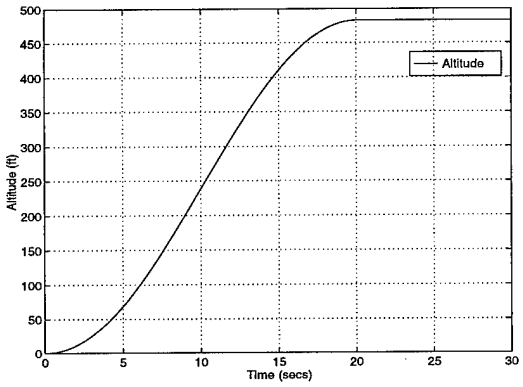

Figure 33. Case 9 - Altitude Output

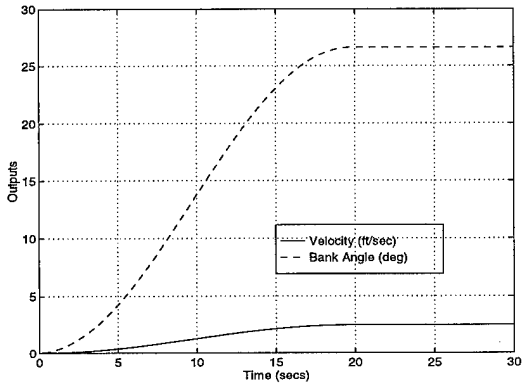

Figure 34. Case 9 - Velocity and Bank Angle Outputs

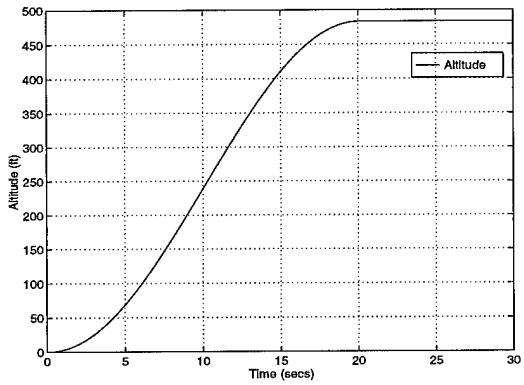

Figure 35. Case 10 - Altitude Output

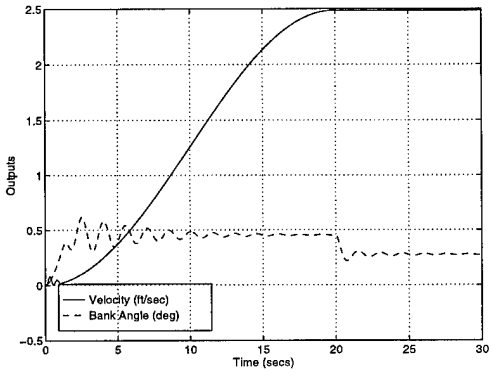

Figure 36. Case 10 - Velocity and Bank Angle Outputs

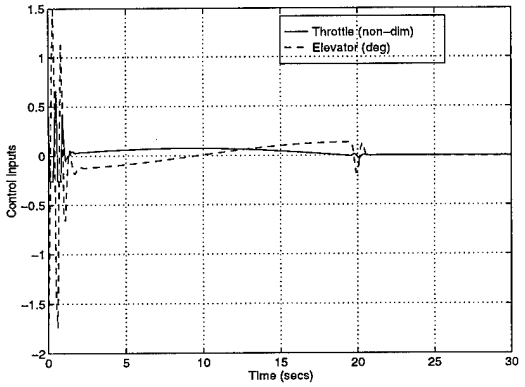

Figure 37. Case 9 - Longitudinal Controls

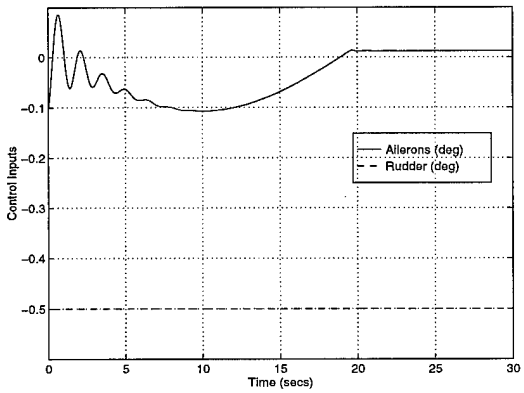

Figure 38. Case 9 - Lateral Controls

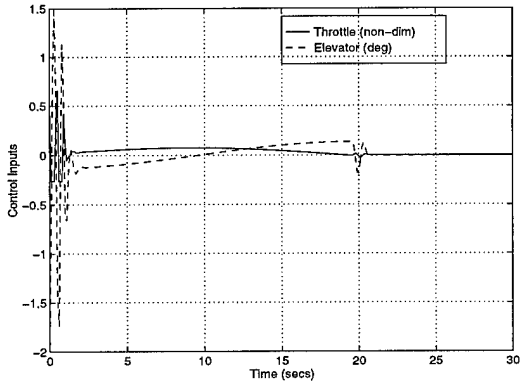

Figure 39. Case 10 - Longitudinal Controls

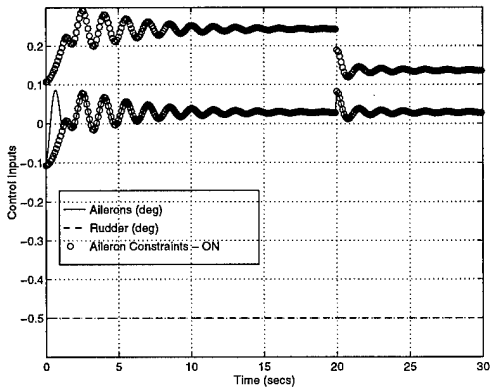

Figure 40. Case 10 - Lateral Controls

4.2.3 Combined Dynamic Response

The final two cases of the linear simulations, listed in Table 13, show the ability of the controller to handle lateral and longitudinal dynamics simultaneously. Since the linear model is nearly decoupled into longitudinal and lateral motion, the resulting outputs of Case 11 were basically the results of Cases 5 and 10 superimposed. Case 12 was the result of the superposition of Cases 7 and 10.

Table 13. Linear Model,Combined Dynamic Cases

Case	A*	B*	C*	D*	E*
11	On	0.75	$\pm 0.5°$	$60°$ Sin Input, $t_{ss} = 10sec$	$500ft$ Sin Input, $t_{ss} = 20sec$
12	On	0.75	$\pm 0.5°$	$60°$ Cos Input, $t_{ss} = 10sec$	$500ft$ Cos Input, $t_{ss} = 20sec$
*A - Aileron Adative Constraint On or Off					
*B - $A + BF_r$ and $A + LC$ Pole Starting Location					
*C - Rudder Stops					
*D - Bank Angle Trajectory					
*E - Altitude Trajectory					

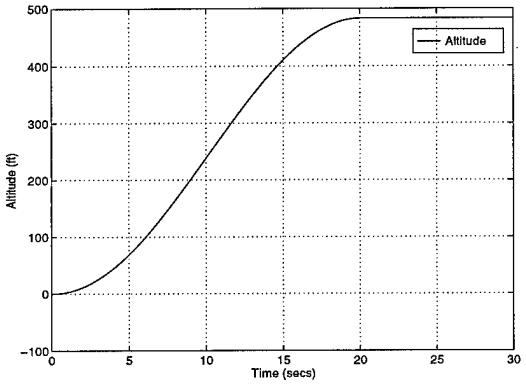

Figure 41. Combined Case 11 - Bank Angle Output

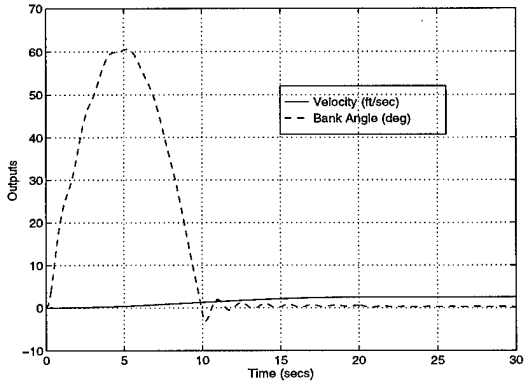

Figure 42. Combined Case 11 - Velocity and Altitude Outputs

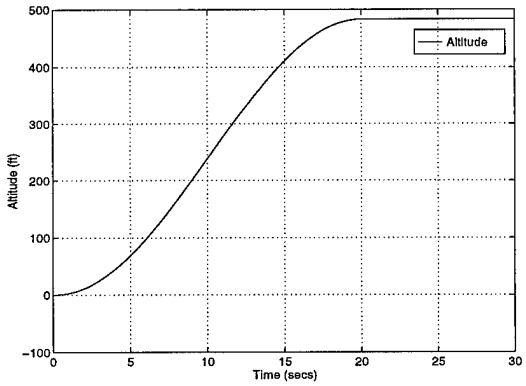

Figure 43. Combined Case 12 - Bank Angle Output

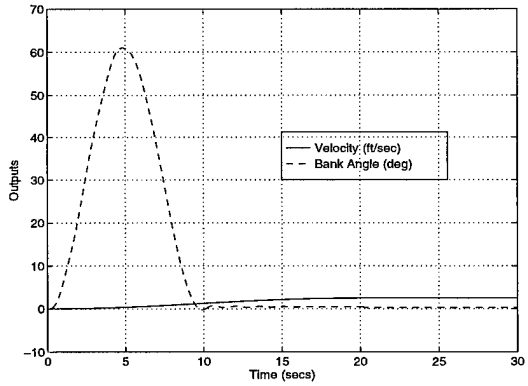

Figure 44. Combined Case 12 - Velocity and Altitude Outputs

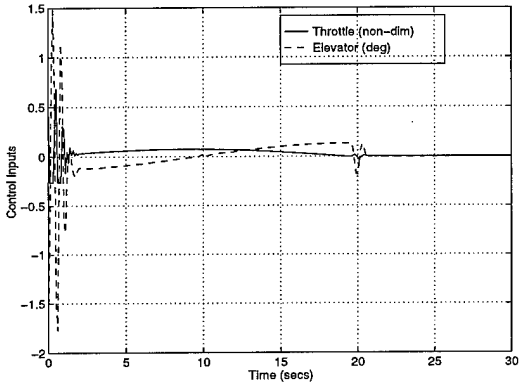

Figure 45. Combined Case 11 - Longitudinal Controls

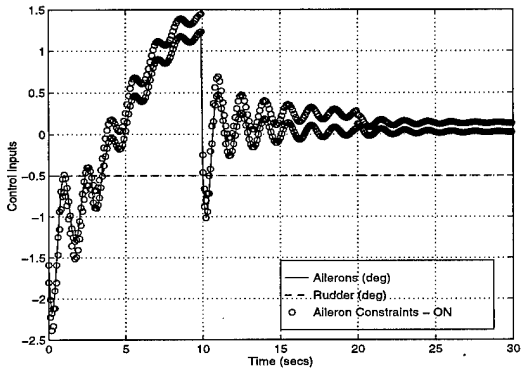

Figure 46. Combined Case 11 - Lateral Controls

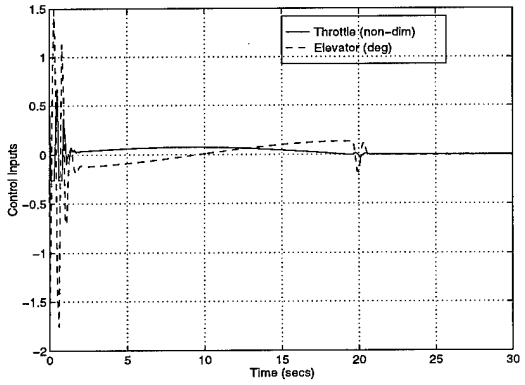

Figure 47. Combined Case12 - Longitudinal Controls

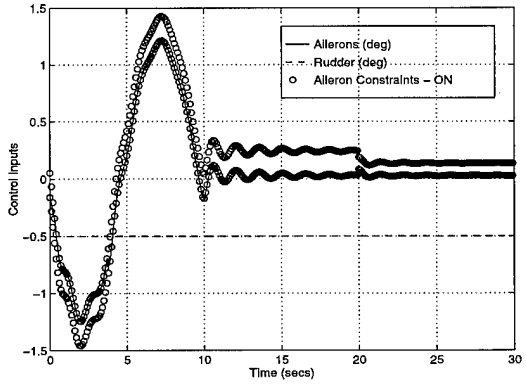

Figure 48. Combined Case 12 - Lateral Controls

4.3 Comparison of Linear and Nonlinear Models

Before proceeding with the nonlinear F-16 model, it was desired to first ensure that the nonlinear model, Figure 2, was working correctly and identify any potential problem areas of using a linearized model to develop the various matrices of the MPC strategy. To achieve these objectives, the linearized F-16 model of Figure 1 was placed in the loop of the MPC controller, Figure 7. The system was commanded according to equation 90, where the output vector $y(k)$ is defined in Equation 91.

$$s = \begin{bmatrix} 5 & 2 & 0 & 0 & 0 \end{bmatrix} \tag{90}$$

$$y \equiv [\ \phi \quad h \quad V \quad \beta \quad \theta\ ] \tag{91}$$

The control inputs to both the linear and nonlinear systems are shown in Figures 49 and 50. As seen in Figure 51 there is excellent correlation between the linear and nonlinear model for bank angle, ϕ. Also there was good agreement of the side slip angle, β, response in Figure 52 and the pitch rate, $\dot{\theta}$, response in Figure 53. However the two areas of concern are with altitude and velocity responses shown in Figures 55 and 54. These results caused concern as the state prediction would be predicting the incorrect future values of velocity and altitude which are fed into the optimizer. The robustness of the MPC strategy would be questioned. A comparison was also made with the c.g. location at 35%. These results are in Appendix D and show worse correlation between altitude and velocity.

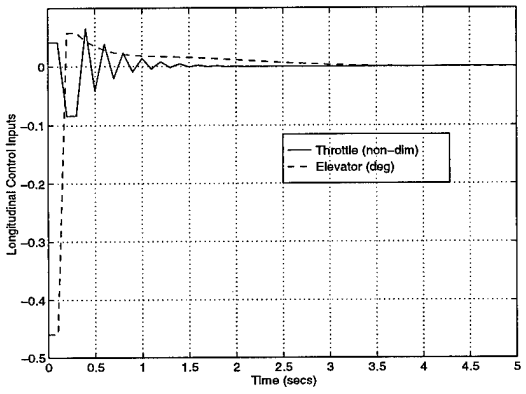

Figure 49. Linear and Nonlinear Comparison - Longitudinal Control Inputs

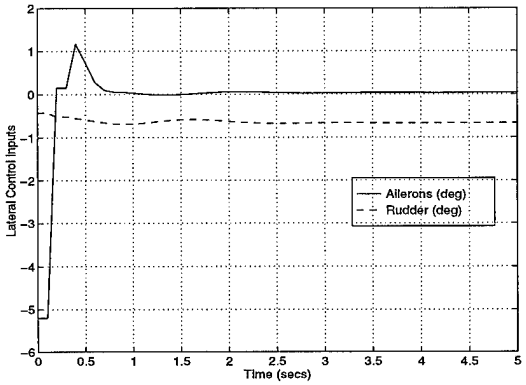

Figure 50. Linear and Nonlinear Comparison - Lateral Control Inputs

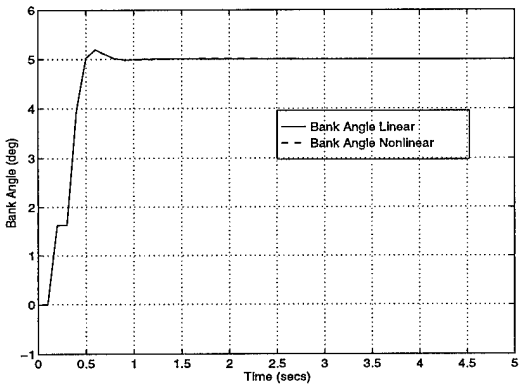

Figure 51. Linear and Nonlinear Comparison - Bank Angle

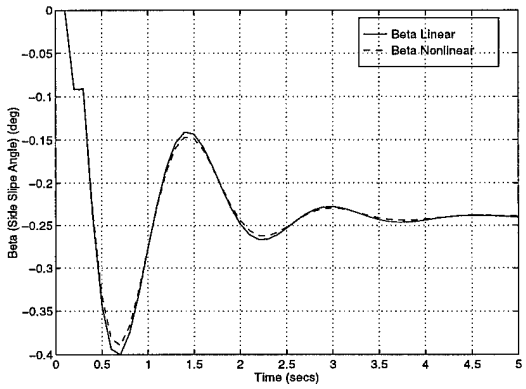

Figure 52. Linear and Nonlinear Comparison - Side Slip Angle

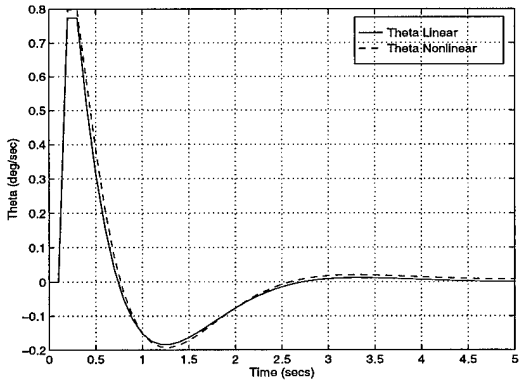

Figure 53. Linear and Nonlinear Comparison - Pitch Rate

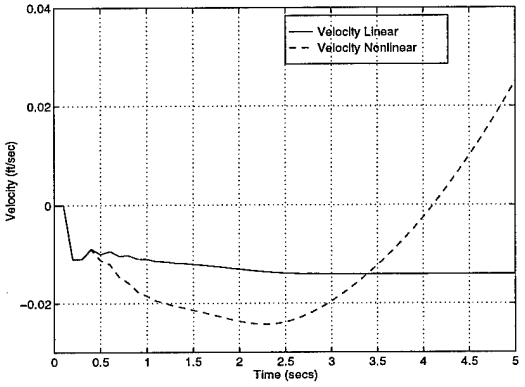

Figure 54. Linear and Nonlinear Comparison - Velocity

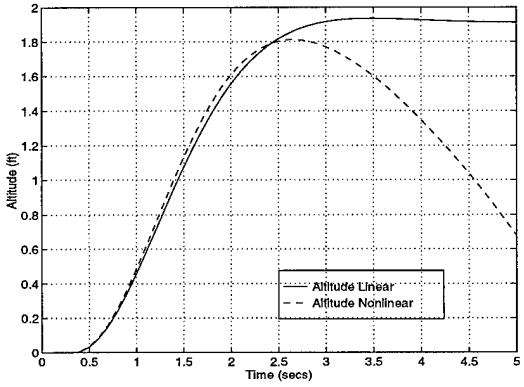

Figure 55. Linear and Nonlinear Comparison - Altitude

4.4 Nonlinear Model Simulation Results

As with the linear simulations, the nonlinear simulations followed a similar path for validating that the MPC controller works on both lateral and longitudinal trajectories as well as combined trajectories. The time step, weighting matrices and output vector used for all nonlinear simulations are given in Equation 92.

$$
\begin{aligned}
T_s &= 0.1\ sec \\
R_u &= \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 1e4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix} \\
R_y &= \begin{bmatrix} 1e2 & 0 & 0 & 0 & 0 \\ 0 & 1e4 & 0 & 0 & 0 \\ 0 & 0 & 1e4 & 0 & 0 \\ 0 & 0 & 0 & 1e2 & 0 \\ 0 & 0 & 0 & 0 & 1e4 \end{bmatrix} \\
y &= \begin{bmatrix} \phi & h & V & \beta & \theta \end{bmatrix}^T
\end{aligned}
\tag{92}
$$

The added outputs, β and θ, were used to assist in system tracking. By adding these two outputs, two additional tuning parameters were added to the MPC problem, specifically their corresponding

elements in the weighting matrix R_y. The indirect benefits of these states were seen in the ability to reduce aileron and elevator oscillations without adversely affecting system tracking.

4.4.1 Bank Angle Response

Building upon the linear results, 5 different nonlinear bank angle response cases were simulated. These cases, listed in Table 14, were used as a basis of comparing the linear and nonlinear responses. In general, the nonlinear bank angle responses closely resembled the linear responses for the same set of conditions. This was expected as the comparison between the linear and nonlinear models showed excellent correlation of the lateral outputs.

Table 14. Nonlinear Model, Bank Angle Cases

Case	A*	B*	C*	D*
13	Off	0.75	$\pm 0.5°$	60° Step Input
14	Off	0.75	$\pm 0.5°$	60° Sin Input, $t_{ss} = 10sec$
15	On	0.75	$\pm 0.5°$	60° Sin Input, $t_{ss} = 10sec$
16	Off	0.75	$\pm 0.5°$	60° Cos Input, $t_{ss} = 10sec$
17	On	0.75	$\pm 0.5°$	60° Cos Input, $t_{ss} = 10sec$
*A - Aileron Adative Constraint On or Off				
*B - $A + BF_r$ and $A + LC$ Pole Starting Location				
*C - Rudder Stops				
*D - Trajectory				

Case 13 demanded a 60° bank step trajectory. Comparing Figure 56 with the linear case, Figure 15, the two bank angle output trajectories are very similar. Even the lateral control inputs, Figures 59 and 20, show similar responses. The only difference is the nonlinear case took approximately twice the time to reach steady state. This is most likely due to aerodynamic effects not accounted for in the LTI model. An interesting result of the nonlinear simulation was the steady state elevator and throttle inputs shown in Figure 58. As discussed in the linear response section, the steady state longitudinal control inputs were zero. Here the controller is simulating a real world response in which additional elevator and throttle inputs are required to generate the extra lift and overcome the additional drag generated in a steady level turn. While the controller did not do a great job in

maintaining zero altitude loss shown in Figure 56, a $40ft$ drop in altitude is realistic in actual flight for a sudden 60° bank angle input. Correct modification of the weighting matrices or employing an adaptive weighting matrix scheme should reduce this altitude deviation.

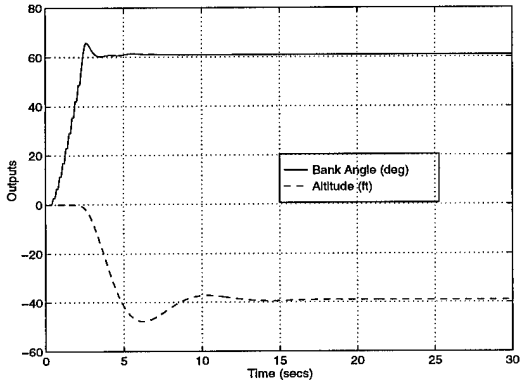

Figure 56. Case 13 - Bank Angle and Altitude

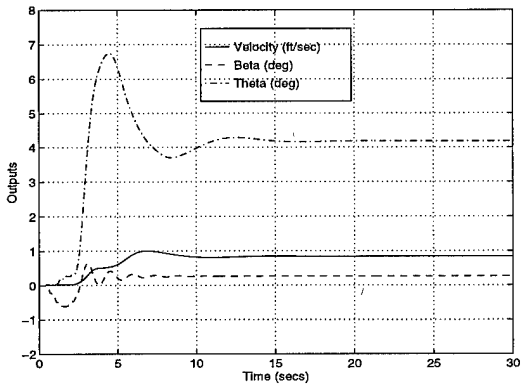

Figure 57. Case 13 -Velocity, Beta and Theta

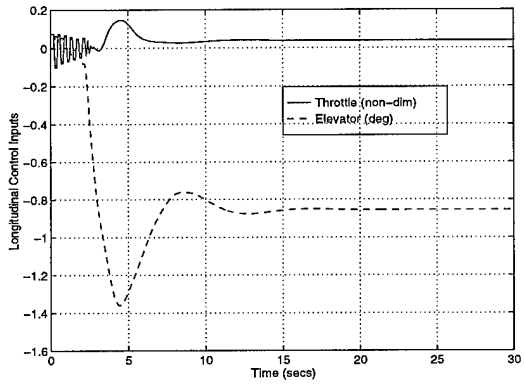

Figure 58. Case 13 - Longitudinal Control Inputs

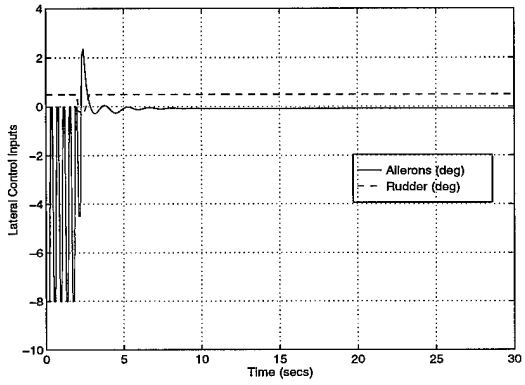

Figure 59. Case 13 - Lateral Control Inputs

Simulation results for Cases 14, 15 and 16 are shown in Appendix D. For Case 14 the controller smoothly followed the sine wave bank trajectory, but lost 40 ft in altitude during the dynamic portion. Upon reaching zero bank angle, the altitude output also returned to zero. It is suspected, as discussed earlier, that the inability of the controller to accurately predict velocity and altitude far future outputs is the cause of the altitude deviation. Both Cases 15 and 16 experienced similar altitude changes before returning to zero deviation. Once the output trajectory had returned to zero

steady state, the nonlinear model outputs and control outputs also smoothly returned to zero. In Case 15, as discovered in the linear Case 5, when adaptive aileron constraints were utilized, there was not as smooth of tracking for the same commanded trajectory as Case 14. Aileron inputs were also oscillatory for Case 15 but followed the general trend as Case 14.

In Case 16 the bank angle response was similar to the results of the linear Case 6. Upon reaching the zero steady state value at 10 *sec* the overshoot was negligible especially when compared to the sin wave trajectory of Case 14. This reduced overshoot was attributed to the boundary conditions of a cosine wave as discussed in the linear bank angle results section. Simulation results for Case 17 are shown in Figures 60 through 63. As expected from earlier results the trajectory is not as smooth since adaptive aileron constraints are being used. However there is still acceptable trajectory following and the altitude deviation is slightly less than Cases 14 through 16.

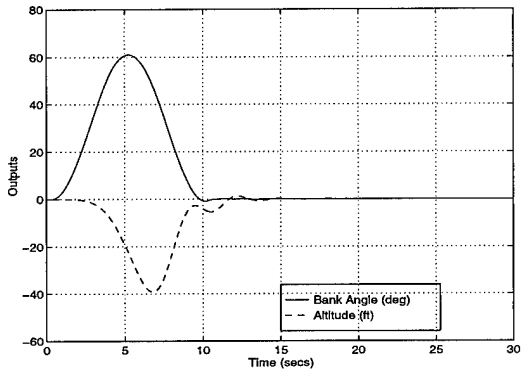

Figure 60. Case 17 - Bank Angle and Altitude

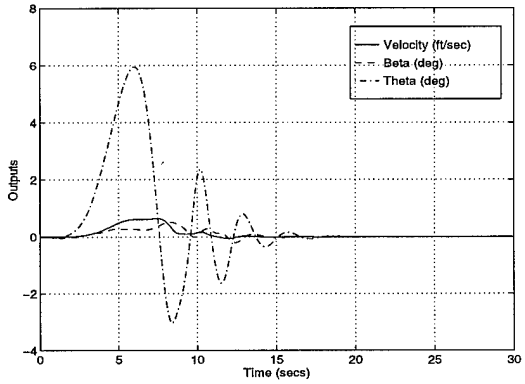

Figure 61. Case 17 - Velocity, Beta and Theta

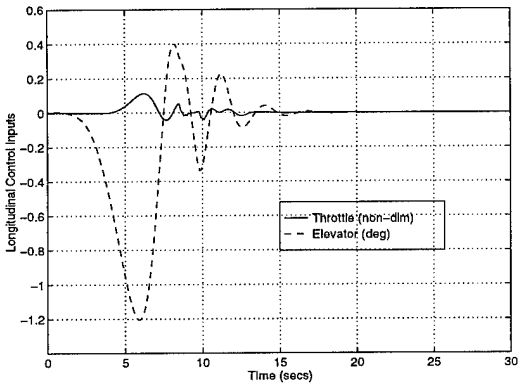

Figure 62. Case 17 - Longitudinal Control Inputs

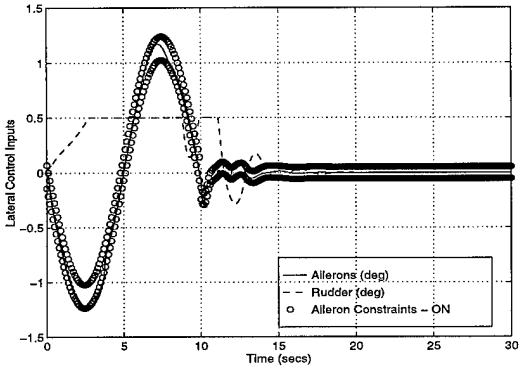

Figure 63. Case 17 - Lateral Control Inputs

4.4.2 Altitude Response

A series of altitude commanded trajectories were simulated as listed in Table 15. Cases 18 through 20 were run to compare the nonlinear response to the linear results. Based upon the success of the Cases 18 - 20, a more demanding altitude change was simulated as seen in Cases 21 through 23. Throughout the simulations restrictive rudder constraints have had either neutral or slightly positive benefits in tracking and reduced control power. Because of the more demanding longitudinal maneuvers, the rudder constraints were further limited to $\pm 0.1°$. Also the placement of the state estimator poles with $L_i = 0.70$, appears to have assisted in the altitude tracking. However quantifiable results using $L_i = 0.70$ were not identified.

Table 15. Nonlinear Model, Altitude Cases

Case	A*	B*	C*	D*
18	Off	0.75 and 0.75	$\pm 0.5°$	$500ft$ Step Input
19	Off	0.75 and 0.75	$\pm 0.5°$	$500ft$ Cos Input, $t_{ss} = 20sec$
20	On	0.75 and 0.75	$\pm 0.5°$	$500ft$ Cos Input, $t_{ss} = 20sec$
21	Off/Off	0.75 and 0.70	$\pm 0.1°$	$9000ft$ Cos Input, $t_{ss} = 60sec$
22	On/Off	0.75 and 0.70	$\pm 0.1°$	$9000ft$ Cos Input, $t_{ss} = 60sec$
23	On/On	0.75 and 0.70	$\pm 0.1°$	$9000ft$ Cos Input, $t_{ss} = 60sec$
*A - Aileron Adative Constraint On or Off				
*B - $A + BF_r$ and $A + LC$ Pole Starting Location				
*C - Rudder Stops				
*D - Trajectory				

Due to instability seen in the linear case of a step altitude commanded trajectory Figure 29, it was informative to see the response of the nonlinear model, Figures 64 through 68 . The nonlinear model exhibited much better system tracking as compared to Case 8. In the nonlinear case, the target altitude of 500 ft was not reached until ~ 7 sec as compared to 3 sec in the linear simulation. It is expected that this 'sluggishness' in the nonlinear model actually increases the likelihood of reaching a steady state value. This 'sluggishness' is due to a lack of understanding of the limits of various accelerations of the nonlinear model and was highlighted in other simulations not included here. For example, a 1000 ft cosine altitude trajectory with $t_{ss} = 6$ sec was commanded of the LTI model. While the commanded rate of climb only reached ~ 250 ft/sec meeting the $\dot{h}$ and throttle requirements of Chapter 3, there was no consideration of the aircraft's acceleration response. As a result the nonlinear model went unstable. The elevator ratcheting seen in Figure 67 was suspected to be a result of both an unrealistic step trajectory and the unconstrained control inputs $u(k + l)$ as discussed in the linear results section.

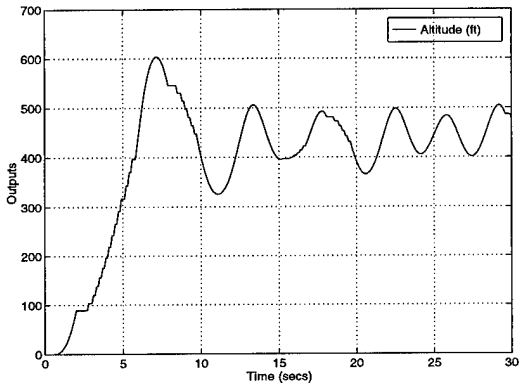

Figure 64. Case 18 - Altitude Output

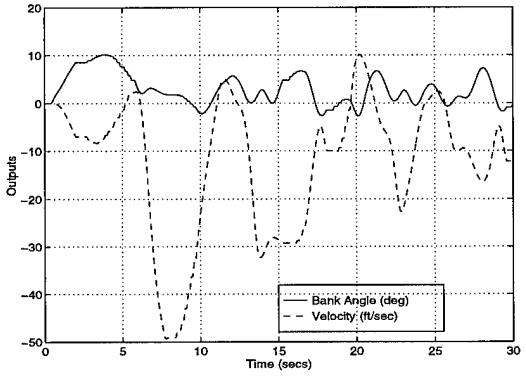

Figure 65. Case 18 - Velocity and Bank Angle Outputs

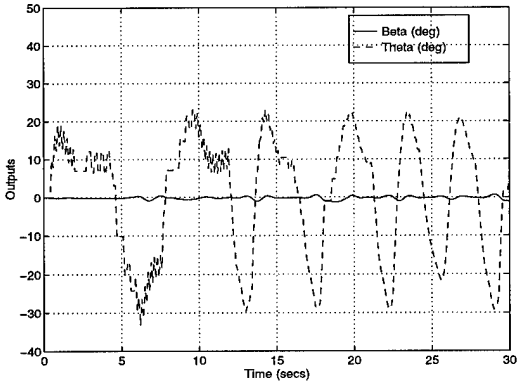

Figure 66. Case 18 - Beta and Theta Outputs

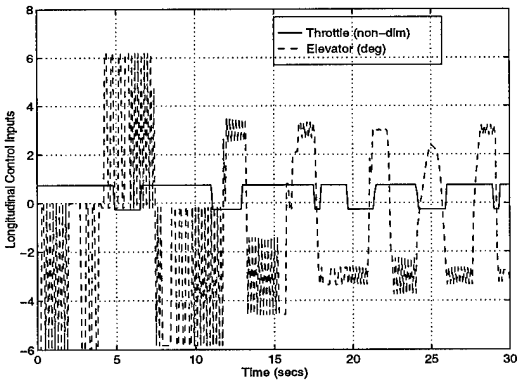

Figure 67. Case 18 - Longitudinal Controls

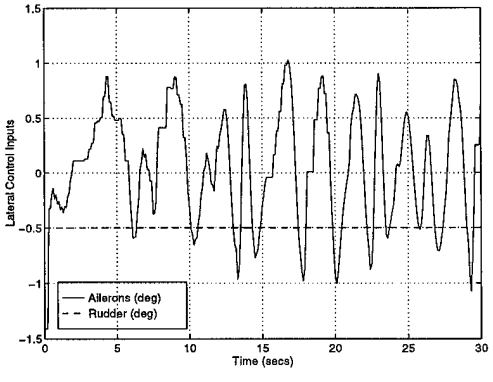

Figure 68. Case 18 - Lateral Controls

Case 19 (see Appendix D) and Case 20, Figures 69 through 73, have near identical model output and control input results. The exception is that the use of adaptive aileron constraints in Case 20, proved to reduce bank angle output deviation by more than 50% over the first 20 *sec.* In both cases though steady state values of bank angle were both $\sim 0.4°$.

As discussed in the linear bank angle results section, there was no stability analysis made on the adaptive aileron constraint technique. This proved to be important because the use of the technique in Equation 80 of Chapter 3 proved to cause the output of Case 20 to initially become unstable. To regain lateral stability the buffer term, $\frac{0.25}{100} m_{max_a}$, which was added to the $\dot{\phi}_e$ term was increased. This provided the optimizer more latitude in selecting a stabilizing aileron input. Equation 93 was the original form

$$\begin{aligned} m'_{max_a} &= \frac{-0.06}{2}\dot{\phi}_e + \frac{0.25}{100} m_{max_a} \\ m'_{min_a} &= \frac{-0.06}{2}\dot{\phi}_e + \frac{0.25}{100} m_{min_a} \end{aligned} \tag{93}$$

and Equation 94 has the increased buffer.

$$\begin{aligned} m'_{max_a} &= \frac{-0.06}{2}\dot{\phi}_e + \frac{0.5}{100} m_{max_a} \\ m'_{min_a} &= \frac{-0.06}{2}\dot{\phi}_e + \frac{0.5}{100} m_{min_a} \end{aligned} \tag{94}$$

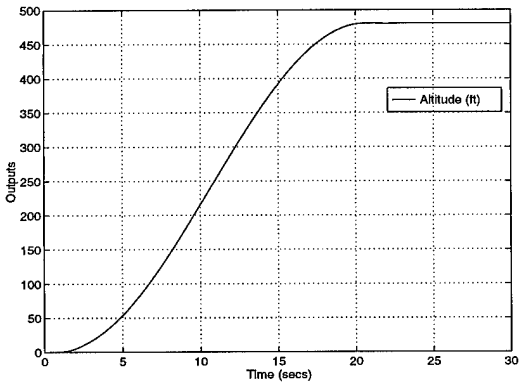

Figure 69. Case 20 - Altitude Output

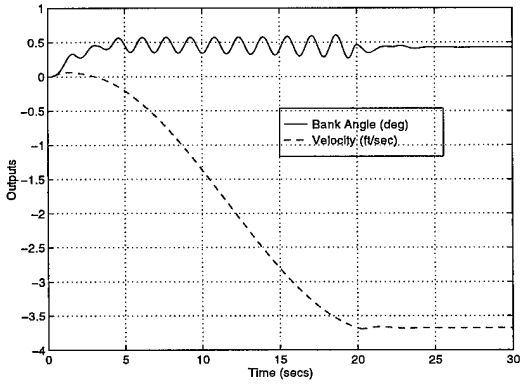

Figure 70. Case 20 - Velocity and Bank Angle Outputs

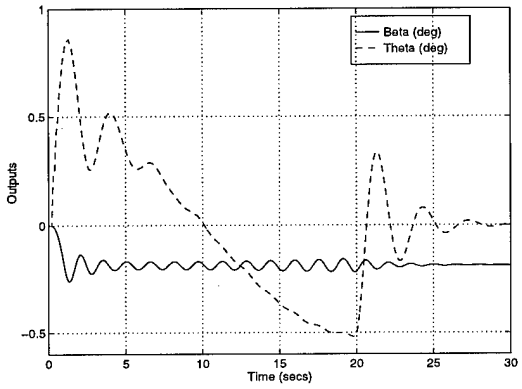

Figure 71. Case 20 - Beta and Theta Outputs

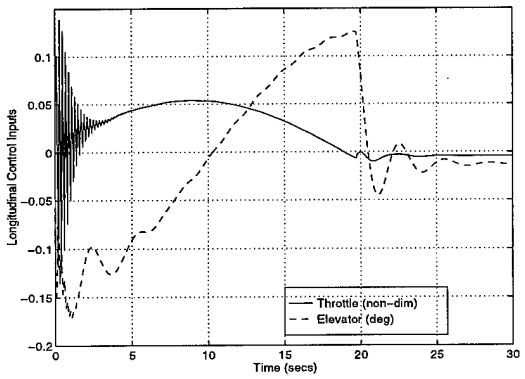

Figure 72. Case 20 - Longitudinal Controls

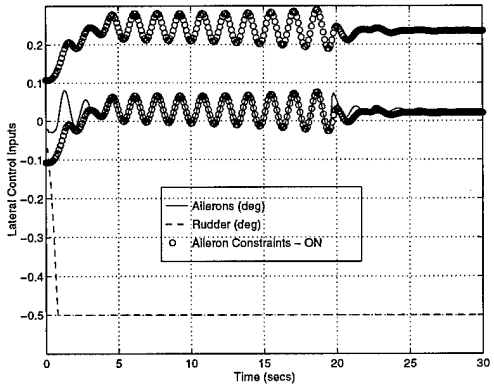

Figure 73. Case 20 - Lateral Controls

The results of Cases 21 through 23 are very similar to each other. Plots for Cases 21 and 22 are found in Appendix D and for Case 23 Figures 74 through 78 are seen below. The use of the adaptive elevator constraint developed in Equation 84 of Chapter 3 had no apparent affect on the results of the simulations. However the use of the adaptive aileron constraint technique was beneficial for ensuring bank angle deviation remained near zero. In Case 21 the bank angle reached a maximum of 10° and a steady state value of 8°. In Cases 22 and 23 the maximum bank angle was less than 1° as well as the steady state value. As discussed and shown previously the only draw back to using adaptive aileron constraints was an increase in aileron control power due to oscillations. However

as seen in Figure 78 the oscillations are negligible when compared to the absolute aileron control inputs of $\pm 21.5°$.

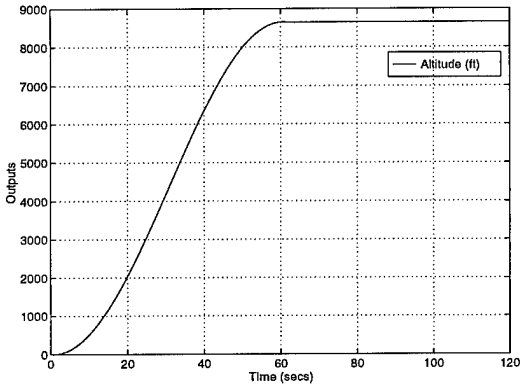

Figure 74. Case 23 - Altitude Output

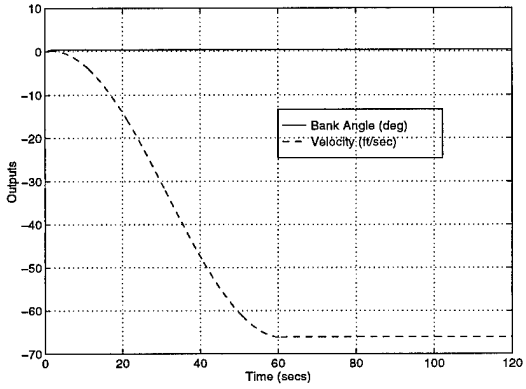

Figure 75. Case 23 - Velocity and Bank Angle Outputs

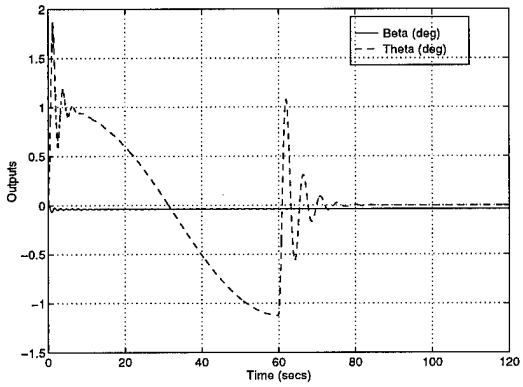

Figure 76. Case 23 - Beta and Theta Outputs

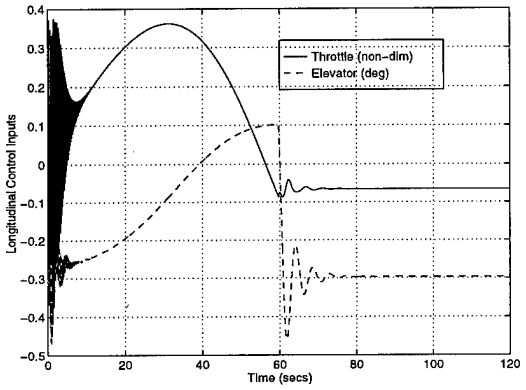

Figure 77. Case 23 -Longitudinal Inputs

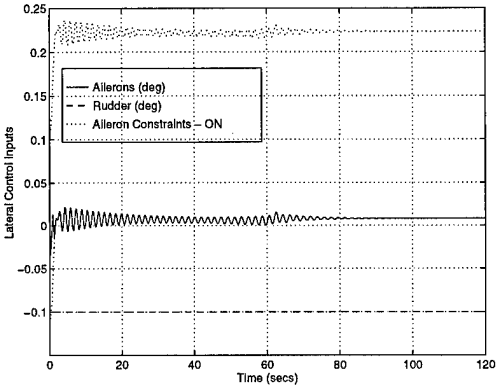

Figure 78. Case 23 - Lateral Inputs

4.4.3 Combined

The simulation cases listed in Table 16 were run to meet the of the initial objective of this thesis. That was to show the MPC strategy can handle combined lateral and longitudinal dynamics using a nonlinear model. Cases 24 and 25 were simulated to compare the affects of adaptive aileron constraints during the combined lateral and longitudinal maneuvers. Case 25 was the nonlinear equivalent to the linear Case 12. Cases 26 through 28 were run to demonstrate the MPC's robustness in handling commanded trajectories which are realistic of actual aerospace systems.

Table 16. Linear Model,Combined Dynamic Cases

Case	**A***	**B***	**C***	**D***	**E***
24	Off/Off	0.75 and 0.75	$\pm 0.5°$	$60°$ Cos Input, $t_{ss} = 10sec$	$500ft$ Cos Input, $t_{ss} = 20sec$
25	On/Off	0.75 and 0.75	$\pm 0.5°$	$60°$ Cos Input, $t_{ss} = 10sec$	$500ft$ Cos Input, $t_{ss} = 20sec$
26	Off/Off	0.75 and 0.70	$\pm 0.1°$	$60°$ Cos Input, $t_{ss} = 20sec$	$9000ft$ Cos Input, $t_{ss} = 60sec$
27	On/Off	0.75 and 0.70	$\pm 0.1°$	$60°$ Cos Input, $t_{ss} = 20sec$	$9000ft$ Cos Input, $t_{ss} = 60sec$
28	On/On	0.75 and 0.70	$\pm 0.1°$	$60°$ Cos Input, $t_{ss} = 20sec$	$9000ft$ Cos Input, $t_{ss} = 60sec$
*A - Aileron Adative Constraint On or Off					
*B - $A + BF_r$and $A + LC$ Pole Starting Location					
*C - Rudder Stops					
*D - Bank Angle Trajectory					
*E - Altitude Trajectory					

The results of Case 24 (Appendix D) are similar to Case 25 results shown in Figures 79 through 83. In both cases altitude response lagged while the bank angle was near its maximum value. With correct initial weighting or even an adaptive weighting scheme, this problem should be solved. The only difference between the two cases was the usual aileron input oscillations caused by the adaptive aileron constraints. The adaptive aileron constraints did not provide any beneficial results in Case 25 and small increase in control power was required as compared to Case 24.

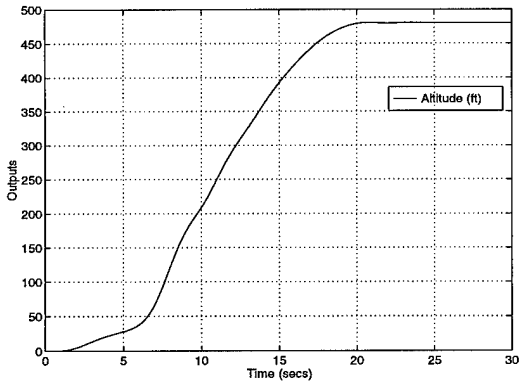

Figure 79. Case 25 - Altitude Output

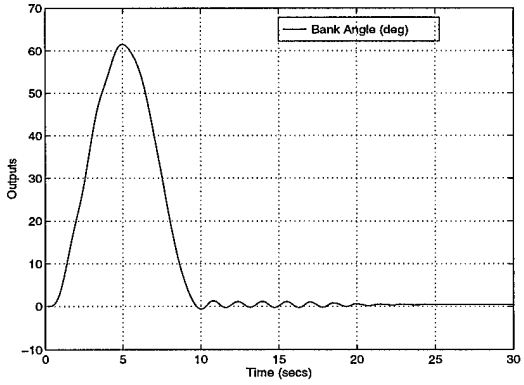

Figure 80. Case 25 - Bank Angle Output

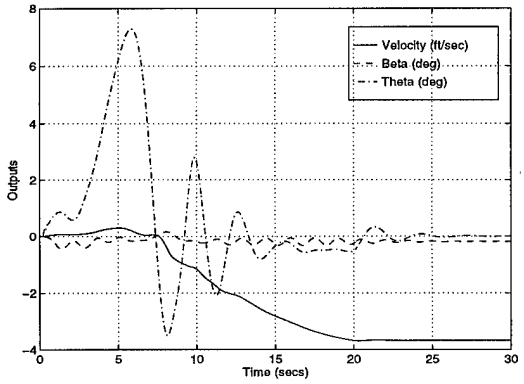

Figure 81. Case 25 - Velocity, Beta and Theta Outputs

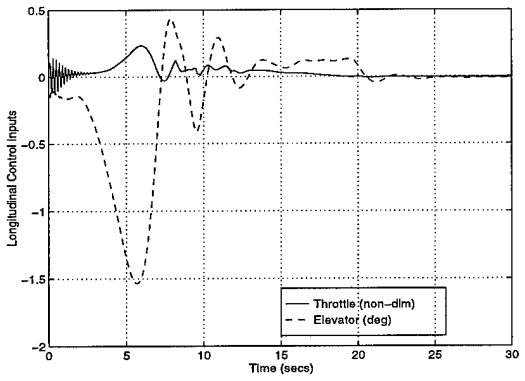

Figure 82. Case 25 - Longitudinal Controls

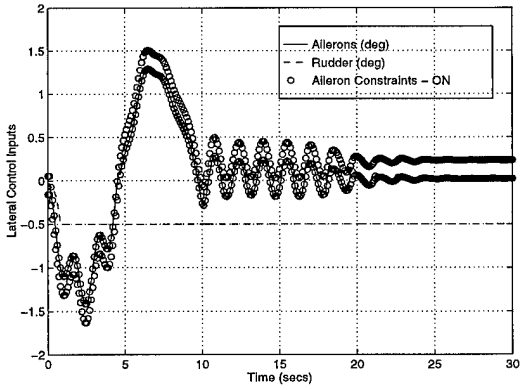

Figure 83. Case 25 - Lateral Controls

However, the use of adaptive aileron constraints was beneficial for system tracking as more demanding trajectories were commanded. In Case 26 (results are in Appendix D) bank angle tracking was smoother than in Figure 85 of Case 27 and the control input oscillations seen in Figure 88 were not experienced in Case 26. But, the steady state bank angle in Case 26 was 8° and it was less than 1° in Case 27. The other outputs and inputs shown in Figures 84 through 88 were nearly identical between the two cases. While the altitude goal of $9000ft$ was not achieved, its steady state error was less than 3.5%. With additional work this error should be reduced. Case 28 had similar results to Case 27 as seen in Figures 89 through 93. The difference between the two cases was the use of elevator adaptive constraints. Figures 87 and 92 show the adaptive elevator constraints truncated the elevator input at 10 *sec.* Despite this truncation there was no apparent affect on lateral or longitudinal outputs. The truncation problem should be overcome by simply readdressing the elevator adaptive constraint technique and including a lateral contribution as discussed in Chapter 3. The elevator constraint technique was included in this thesis to demonstrate it had no adverse affect on system tracking.

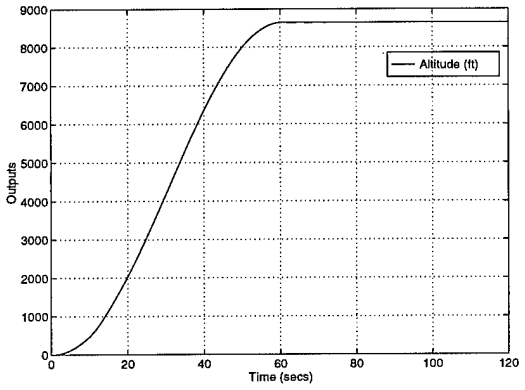

Figure 84. Case 27 - Altitude Output

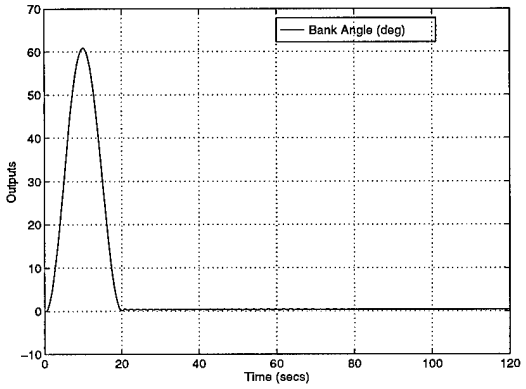

Figure 85. Case 27 - Bank Angle Output

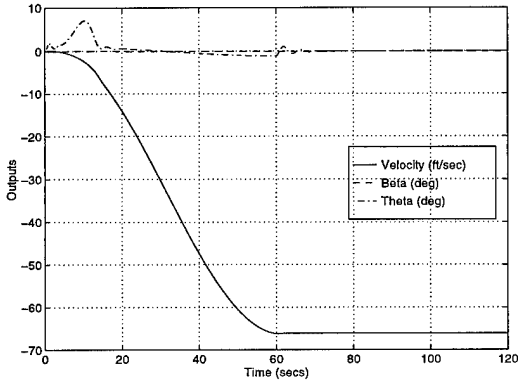

Figure 86. Case 27 - Velocity, Beta and Theta Outputs

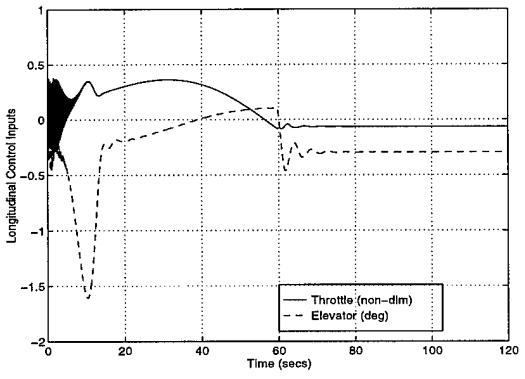

Figure 87. Case 27 - Longitudinal Controls

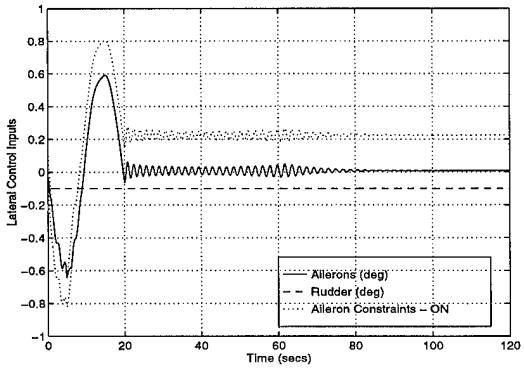

Figure 88. Case 27 - Lateral Controls

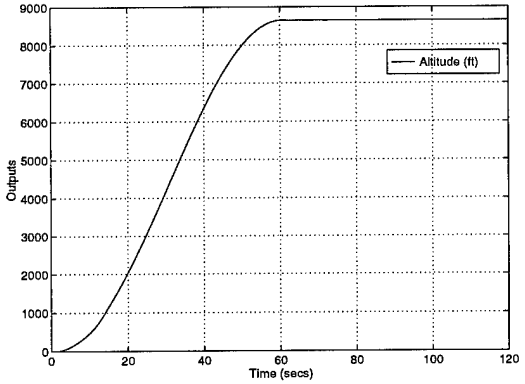

Figure 89. Case 28 - Altitude Output

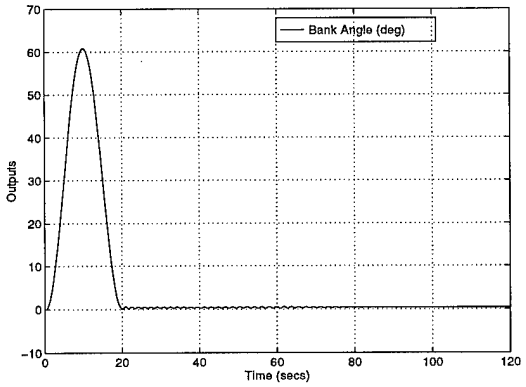

Figure 90. Case28 - Bank Angle Outputs

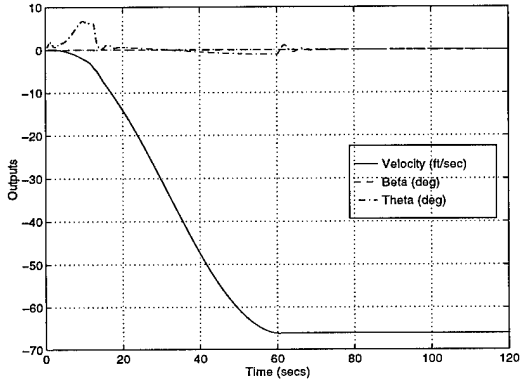

Figure 91. Case 28 - Velocity, Beta and Theta Outputs

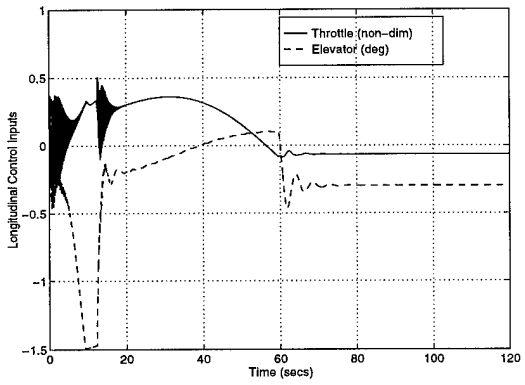

Figure 92. Case 28 - Longitudinal Controls

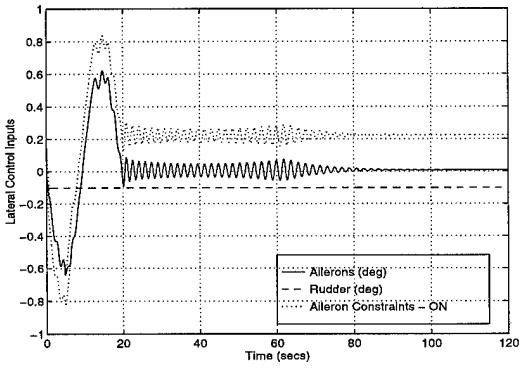

Figure 93. Case 28 - Lateral Controls

Another unique feature of Cases 26 through 28 was that the final location of the controller poles had moved significantly. Equation 95 lists the starting and ending locations of the poles.

$$
\begin{aligned}
eig(A+BF_r)_{start} &= [\ 0.750 \quad 0.746 \quad 0.742 \quad 0.737 \quad 0.733 \quad 0.729 \quad \ldots \\
&\quad\ 0.725 \quad 0.721 \quad 0.717 \quad 0.713\ ] \\
eig(A+BF_r)_{end} &= [\ 0.886 \pm 0.0853i \quad 0.858 \quad 0.795 \pm 0.127i \quad 0.795 \pm 0.089i \quad \ldots \\
&\quad\ 0.718 \pm 0.241i \quad 0.618\ ] \\
eig(A+LC)_{start} &= [\ 0.700 \quad 0.696 \quad 0.692 \quad 0.688 \quad 0.684 \quad 0.680 \quad \ldots \\
&\quad\ 0.677 \quad 0.673 \quad 0.669 \quad 0.665\ ] \\
eig(A+LC)_{end} &= [\ 0.919 \quad 0.796 \pm 0.039i \quad 0.730 \quad 0.721 \pm 0.065i \quad \ldots \\
&\quad\ 0.706 \quad 0.617 \quad 0.527 \quad 0.520\ ]
\end{aligned}
\tag{95}
$$

Chapter 5 - Conclusions and Future Research

5.1 Conclusions

As shown, the original objectives of using the MPC strategy were achieved. Numerous simulations demonstrated the ability of the MPC strategy to handle dynamic trajectories while using a nonlinear model. Also, the placement of the controller poles at the origin to guarantee system tracking was relaxed to a simply stable requirement with a heuristic guarantee of a finite error using a linear model. Finally two adaptive constraint techniques were developed and simulated. The aileron adaptive constraint technique successfully demonstrated the ability to reduce lateral steady state error with minimal increases in required control power. The adaptive elevator technique was not analyzed in depth and showed only neutral effects on tracking and control power usage.

Throughout the research, an attempt was made to identify the influence of the various tuning parameters listed in Chapter 2 on system tracking and required control power. However, generalizations regarding the affects of these tuning parameters on system performance were not developed because a detailed sensitivity analysis was not performed. Additionally the influence of the tuning parameters on the system performance was suspected to be nonlinear. This nonlinear behavior was demonstrated through various simulations as individual changes were made in the placement of controller poles, weighting matrix elements, time step selection and dynamic trajectory design.

Several simulations were performed in which controller poles were placed either closer to or further away from the origin. However there appeared to be no consistency in improved system tracking or reduced control power. In general, there was an inverse relationship between elements of the weighting matrices, R_y and R_u, provided there was a strong coupling of the corresponding inputs and outputs (i.e. aileron deflection and roll rate). However there were cases where this relationship did not hold true. Selection of the time step, T_s, also demonstrated nonlinear effects. Reduction of

the time step from 0.1 *sec* to less than 0.05 *sec* generally made the system unstable. Increasing the time step also produced unstable results. Dynamic trajectory design was also an important factor in achieving stable simulations. Various polynomial bank angle and altitude trajectories were designed with smooth boundary conditions, but none were as successful as the sine and cosine trajectories. Thus more complex trajectory design might be required to consist of a sum of a finite number of sine and cosine functions. Also for dynamic trajectory design, it is important to consider the physical limits of the aircraft. For example, a trajectory was designed where the rate of climb, $\dot{h}$, fell within the bounds of the nonlinear model. However the derivative of $\dot{h}$, was beyond the capability of the nonlinear model and caused the simulation of the entire closed loop system to become unstable.

5.2 Future Research/Projects

With the eventual goal of realizing the MPC strategy on an actual aircraft, five potential research topics not covered in this thesis are presented. First, as was seen in Chapter 4, linear and nonlinear states quickly diverged from each other when provided with the same control inputs. It is expected that a simple nonlinear model which incorporates the effects of lateral motion would solve this problem. The prediction matrices would then be rewritten to incorporate this nonlinear model. A simpler interim solution would base the linear prediction matrices upon an LTI model which was calculated at some small bank angle. However, the problem of lateral and longitudinal coupling would still exist. Second, the design philosophy used in this thesis separated the commanded trajectory code from the optimization code. Doing this allows the incorporation of a pilot model to determine the commanded trajectory. One concept of a simplified pilot model would consist of three parameters. First, longitudinal stick inputs would correspond to pitch angle changes. Second, lateral stick inputs would correspond to roll rate changes. And last throttle inputs would correspond to altitude changes. The third idea consists of developing a gain scheduling technique. Since the dy-

namics of a nonlinear model are constantly changing, the use of gain scheduling could significantly improve system tracking. The design used in this thesis left open the possibility of selecting the prediction matrices as well as the inner loop feedback matrices. The selection criteria could be based upon any combination of outputs, control inputs or their derivatives. The selection would take place at each time step within the optimization routine. Fourth, to simulate more realistic conditions, the SimStar hybrid computer is available here at AFIT. Using this asset, the cumbersome requirement of numerical integration would be performed by the analog portion of the SimStar computer. And finally, perhaps the most difficult of these topics is the identification of the effects of the various tuning parameters on system performance.

APPENDIX A - Matlab Code

A.1 Modgen.m

```
%
% LTI Model Generation for Thesis using Aircraft Control
% and Simulation by Stevens and Lewis
% Capt Christopher M. Shearer
% 1 Jul 97
%
clear
global ay az
xguess = [669 0.01 0 0. 0.01 0 0 0 0 0 0 100 16.98];
xguess = [669.796 0.0111544 0 -6.27372e-10 ...
0.0111544 0 -0 -0 0 0 0 ...
100 16.9845];

uguess = [0.261541 -1.54463 6.18088e-11 -2.02108e-11];
const(1) = 0.0; % const(1) = gamma (deg)
% const(2) = Sin(Gamma)
const(3) = 0.0; % const(3) = Roll Rate (deg/sec)
const(4) = 0.0; % const(4) = Pitch Rate (deg/sec)
const(5) = 0.0; % const(5) = Turn Rate (deg/sec)
const(6) = 0.0; % const(6) = Phi (Bank Angle) (deg)
% const(7) = Cos(Phi)
% const(8) = Sin(Phi)
const(9) = 0.0; % const(9) = Theta dot (deg/sec)
const(10) = 1; % const(10) = Coordnated Turn (1 = yes, 0 = no)
const(11) = 0.0; % const(11) = Rate of Climb (ft/sec)
const(12) = 0.0; % const(12) = Bleed Rate (ft/sec^2)
const(13) = 5 % const(13) = Orient (1 = Wings Level, Steady Flight)
% (2 = Constant Climb Angle, Gamma)
% (3 = Constant Altitude, Constant Turn Rate)
% (4 = Constant Pitch Rate, theta dot)
% (5 = Constant Turn Rate and Constant Climb)
cases = case1
n = size(cases,1);
```

```
for i = 1:n
 mach = cases(i,1)
 alt = ca
ses(i,2);
 TR = cases(i,3);
 RC = cases(i,4);
 fname = ['vm',num2str(mach*10),'a',num2str(alt/100),'tr',...
 num2str(TR*10),'rc',num2str(RC/100),'.out'];
 fid = fopen(fname,'w');
 xguess(12) = alt;
 xguess(1) = adc1(mach,alt);
 const(5) = TR;
 const(11) = RC;
%%%%% Go find steady state values of states %%%%%
 [x,u,fcost,lcost] = trimmer(xguess,uguess,const,fid);
%%%%% Go compute the state space matrices %%%%%
 [a,b,c,d] = jacob1(x,u);
%%%%% Get the X dot terms
 xd = subf16(x,u);
 [amach,qbar]=adc(x(1),x(12));
%%%%% Output the results
 dataout(x,u,const,xd,fcost,lcost,ay,az,qbar,amach,a,b,c,d,fid);
%%%%% Close the current file
 fclose(fid);
%%%%% Set the current steady state values to the guess for the next time
% xguess = x;
% uguess = u;
end
```

A.2 Trimmer.m

```
function [x,u,fcost,lcost]=trimmer(Xguess,Uguess,const,fid)
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program: trimmer
% by: Mech 628 Incredible Group 1
% Ise, Shearer, Clark
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% This program numerically calculates the equilibrium state and control vectors of an F-16 model given
% certain parameters. Inputs include initial guesses for the equilibrium state and input vectors.
% If the routine is called with no inputs the user will be prompted to key the equilibrium initial
% guesses in by hand.
% The user will be prompted to pick one of the following A/C orientation options
% and provide the desired altitude, airspeed, gamma, turn rate, pitch rate,etc. :
%
% 1. Wings Level (gamma = 0)
% 2. Wings Level (gamma <> 0)
% 3. Steady Constant Altitude Turn
% 4. Steady Pull Up
% 5. Steady Turn and Constant Climb
%
% The user will also be prompted for the number of iterations to be used in the numerical
% minimization search.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% states: controls:
% x1 = Vt x4 = phi x7 = p x10 = pn u1 = throttle
% x2 = alpha x5 = theta x8 = q x11 = pe u2 = elevator
% x3 = beta x6 = psi x9 = r x12 = alt u3 = aileron
% x13 = pow u4 = rudder
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Script/Function calls:
% getinput
% subf16
% clf16
% conf16
% fminsa
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
global ay az
format long;
if(nargin==4)
```

```
x=Xguess;
u=Uguess;
else
x=zeros(1,13);
u=zeros(1,4);
end
%
% const(1) = gamma (deg)
% const(2) = Sin(Gamma)
% const(3) = Roll Rate (deg/sec)
% const(4) = Pitch Rate (deg/sec)
% const(5) = Turn Rate (deg/sec)
% const(6) = Phi (Roll Angle) (deg)
% const(7) = Cos(Phi)
% const(8) = Sin(Phi)
% const(9) = Theta dot (deg/sec)
% const(10) = Coordnated Turn (1 = yes, 0 = no)
% const(11) = Rate of Climb
% const(12) = Bleed Rate
% const(13) = Orient (1 = Wings Level, Steady Flight)
% (2 = Constant Climb Angle, Gamma)
% (3 = Constant Altitude, Constant Turn Rate)
% (4 = Constant Pitch Rate, theta dot)
% (5 = Constant Turn Rate and Constant Climb)
%
rtod = 180/pi;
orient = const(13);
ndof = 6;
const(1) = const(1)/rtod;
const(2) = sin(const(1));
const(3) = const(3)/rtod;
const(4) = const(4)/rtod;
const(5) = const(5)/rtod;
const(6) = const(6)/rtod;
const(7) = cos(const(6));
const(8) = sin(const(6));
const(9) = const(9)/rtod;
```

```
clear s
if (orient == 3) | (orient == 5)
 s(1)=u(1);
 s(2)=u(2);
 s(3)=u(3);
 s(4)=u(4);
 s(5)=x(2);
 s(6)=x(4);
 s(7)=x(5);
end
if (orient == 1) | (orient == 2) | (orient == 4)
 s(1)=u(1);
 s(2)=u(2);
 s(3)=x(2);
end
cont = 100.0;
itertot = 0.0;
iter = 1500;
tol = 0.5;
%%%%% Now solve for the trimmed condition as long as the change in
%%%%% the cost function is greater than 0.5 percent
while cont > tol
 options = [0 1.0E-9 1.0E-9 0 0 0 0 0 0 0 0 0 0 iter];
 [s,options,x,u,fcost,lcost] = fminsa('clf16',s,options,[],x,u,const);
 fprintf('Initial Cost Function: %g\n', fcost);
 fprintf('Final Cost Function: %g\n', lcost);
 cont = abs((fcost-lcost)/fcost) * 100
 if lcost < 1.0e-5
 cont = tol*0.9;
 end
 itertot = itertot + iter

end
```

A.3 Jacob.m

```
function [a,b,c,d]=jacob(Xequil,Uequil)
% [a,b,c,d]=jacob(xequil,uequil)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program : Jacob
% by: Mech 628 Incredible Group 2
% Capts Chapa & St. Germain
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This program numerically calculates the linearized A, B, C, & D matrices of an F-16 model given
% certain parameters. Inputs include the equilibrium state vector and input vector. If the routine
% is called with no inputs the user will be prompted to key the equilibrium values in by hand.
% Output may be shown coupled longitudinal/lateral or separate. If convergence is not reached for
% any value, the user is prompted for an estimate.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% states: controls:
% x1 = Vt x4 = phi x7 = p x10 = pn u1 = throttle
% x2 = alpha x5 = theta x8 = q x11 = pe u2 = elevator
% x3 = beta x6 = psi x9 = r x12 = alt u3 = aileron
% x13 = pow u4 = rudder
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Order of the measurements described by y = Cx + Du are:
%
% Longintudinal (Elevator Only): y = [ Az q alpha theta Vt ]T
% Longintudinal (Elevator and Throttle): y = [ Az q alpha theta Vt ]T
% Lateral: y = [ Ay p r beta phi ]T
% Coupled Long & Lat: y = [ Az q alpha theta Vt Ay p r beta phi ]T
%
% Note: angles are in degrees, angular rates are in deg/s,
% airspeed in ft/sec, accelerations (Az, Ay) are non-dimensional (i.e., in g's)
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Order of the inputs u are:
%
% Longintudinal (Elevator Only): u = [ el ]T
% Longintudinal (Elevator and Throttle): u = [ thtl el] T
% Lateral: u = [ ail rdr ]T
% Coupled Long & Lat: u = [ thtl el ail rdr ]T
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Script/Function calls:
% getinput
% subf16
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
global az ay;
if nargin==2
 x=Xequil;
 u=Uequil;
else
 disp(' ')
 disp('Input The Equilibrium State And Control Vectors:')
 disp(' ')
 getinput
end
xe=x;ue=u;
tol=0.0001;
xde=subf16(x,u);
aze=az;aye=ay;
%%%%% A matrix %%%%%
for i=1:13
 for j=1:13
 x=xe;del=0.01;slope1=0;diff=.9;
 if xe(i)==0
 del=0.5;
 else
 del=del*xe(i);
 end
 while diff > tol
 x(i)=xe(i)+del;
 xd=subf16(x,u);
 slope2=(xd(j)-xde(j))/(del);
 diff=abs(slope1-slope2);
 del=del*.1;
 slope1=slope2;
 if diff >1e6
 fprintf('No convergence for perturbed state X1 with state X2 : ');
 X1=i
 X2=j
 slope1=input('Enter estimate : ');
 diff=0;
 end
 end
 AA(j,i)=slope1;
 end
end
```

```
%%%%% B matrix %%%%%
x=xe;
for i=1:4
 for j=1:13
 u=ue;del=0.01;slope1=0;diff=1;
 if ue(i)==0
 del=0.5;
 else
 del=del*ue(i);
 end
 while diff > tol
 u(i)=ue(i)+del;
 xd=subf16(x,u);
 slope2=(xd(j)-xde(j))/(del);
 diff=abs(slope2-slope1);
 del=del*.1;
 slope1=slope2;
 if diff >1e6
 fprintf('No convergence for perturbed input U with state X : ');
 U=i
 X=j
 slope1=input('Enter estimate : ');
 diff=0;
 end
 end
 BB(j,i)=slope1;
 end
end
```

```
%%%%% C matrix %%%%%
u=ue;
for i=1:13 % az
 x=xe;del=0.01;slope1=0;diff=1;
 if xe(i)==0
 del=0.5;
 else
 del=del*xe(i);
 end
 while diff > tol
 x(i)=xe(i)+del;
 xd=subf16(x,u);
 slope2=(az-aze)/(del);
 diff=abs(slope2-slope1);
 del=del*.1;
 slope1=slope2;
 if diff >1e6
 fprintf('No convergence for az with state X : ');
 X=i
 slope1=input('Enter estimate : ');
 diff=0;
 end
 end
 CC(1,i)=slope1/(-32.2);
end
for i=1:13 % ay
 x=xe;del=0.01;slope1=0;diff=1;
 if xe(i)==0
 del=0.5;
 else
 del=del*xe(i);
 end
 while diff > tol
 x(i)=xe(i)+del;
 xd=subf16(x,u);
 slope2=(ay-aye)/(del);
 diff=abs(slope2-slope1);
 del=del*.1;
 slope1=slope2;
 if diff >1e6
 fprintf('No convergence for ay with state X : ');
 X=i
```

```
 slope1=input('Enter estimate : ');
 diff=0;
 end
 end
 CC(6,i)=slope1/(32.2);
end
rtod = 180/pi;
CC(2,:)=[0 0 0 0 0 0 0 rtod 0 0 0 0 0];CC(3,:)=[0 rtod 0 0 0 0 0 0 0 0 0 0 0];
CC(4,:)=[0 0 0 0 rtod 0 0 0 0 0 0 0 0 0];CC(7,:)=[0 0 0 0 0 0 rtod 0 0 0 0 0 0];
CC(8,:)=[0 0 0 0 0 0 0 0 rtod 0 0 0 0];CC(9,:)=[0 0 rtod 0 0 0 0 0 0 0 0 0 0];
CC(10,:)=[0 0 0 rtod 0 0 0 0 0 0 0 0 0];CC(5,:)=[1. 0 0 0 0 0 0 0 0 0 0 0 0 0];
%%%%% D matrix %%%%%
x=xe;
for i=1:4 % az
 u=ue;del=0.01;slope1=0;diff=1;
 if ue(i)==0
 del=0.5;
 else
 del=del*ue(i);
 end
 while diff > tol
 u(i)=ue(i)+del;
 xd=subf16(x,u);
 slope2=(az-aze)/(del);
 diff=abs(slope2-slope1);
 del=del*.1;
 slope1=slope2;
 if diff >1e6
 fprintf('No convergence for az with input U : ');
 U=i
 slope1=input('Enter estimate : ');
 diff=0;
 end
 end
 D(1,i)=slope1/(-32.2);
end
```

```
for i=1:4 % ay
 u=ue;del=0.01;slope1=0;diff=1;
 if ue(i)==0
 del=0.5;
 else
 del=del*ue(i);
 end
 while diff > tol
 u(i)=ue(i)+del;
 xd=subf16(x,u);
 slope2=(ay-aye)/(del);
 diff=abs(slope2-slope1);
 del=del*.1;
 slope1=slope2;
 if diff >1e6
 fprintf('No convergence for ay with input U : ');
 U=i;
 slope1=input('Enter estimate : ');
 diff=0;
 end
 end
 D(6,i)=slope1/(32.2);
end
D(2,:)=[0 0 0 0];D(3,:)=[0 0 0 0];D(4,:)=[0 0 0 0];D(7,:)=[0 0 0 0];
D(8,:)=[0 0 0 0];D(9,:)=[0 0 0 0];D(10,:)=[0 0 0 0];D(5,:)=[0 0 0 0];
%%%%% transform A,B,C to book's format %%%%%

posit=[1 2 5 8 13 3 4 7 9];
for i=1:9
 A(1,i)=AA(1,posit(i));A(2,i)=AA(2,posit(i));A(3,i)=AA(5,posit(i));
 A(4,i)=AA(8,posit(i));A(5,i)=AA(13,posit(i));A(6,i)=AA(3,posit(i));
 A(7,i)=AA(4,posit(i));A(8,i)=AA(7,posit(i));A(9,i)=AA(9,posit(i));
end
B(1,:)=BB(1,:);B(2,:)=BB(2,:);B(3,:)=BB(5,:);B(4,:)=BB(8,:);B(5,:)=BB(13,:);
B(6,:)=BB(3,:);B(7,:)=BB(4,:);B(8,:)=BB(7,:);B(9,:)=BB(9,:);
C(:,1)=CC(:,1);C(:,2)=CC(:,2);C(:,3)=CC(:,5);C(:,4)=CC(:,8);C(:,5)=CC(:,13);
C(:,6)=CC(:,3);C(:,7)=CC(:,4);C(:,8)=CC(:,7);C(:,9)=CC(:,9);
%%%%% output to a,b,c,d %%%%%
```

```
disp(' ')
fprintf('1) Longitudinal set (az,q,alpha,theta) With elevator only for input \n');
fprintf('2) Longitudinal set (az,q,alpha,theta) With elevator and throttle for inputs \n');
fprintf('3) Lateral set (ay,pr,r,beta,phi) \n');
fprintf('4) Combined Lateral and Longitudinal set \n\n');
%choice2=input('Choose your output format : ');
%%%%% set choice2 to 1 for MECH 628 final
choice2 = 4
if choice2==1
 A1=1;A2=4;B1=2;B2=2;C1=1;C2=5;D1=1;D2=5;C3=1;C4=4;D3=2;D4=2;
elseif choice2==2
 A1=1;A2=5;B1=1;B2=2;C1=1;C2=5;D1=1;D2=5;C3=1;C4=5;D3=1;D4=2;
elseif choice2==3
 A1=6;A2=9;B1=3;B2=4;C1=6;C2=10;D1=6;D2=10;C3=6;C4=9;D3=3;D4=4;
else
 A1=1;A2=9;B1=1;B2=4;C1=1;C2=10;D1=1;D2=10;C3=1;C4=9;D3=1;D4=4;
end
a=A(A1:A2,A1:A2);
b=B(A1:A2,B1:B2);
c=C(C1:C2,C3:C4);
d=D(D1:D2,D3:D4);
```

A.4 Runme.m

```
% Orginal author Derek W. Ebdon
% Modified Extensively by Capt Chris Shearer, Jul 97
%
% Script file for controller setup. "setup.m"
% Lateral and Longitudinal motion
% CONTROL SURFACES: All functioning
%
% The initial flight condition is as follows:
% Alt.: 100 ft
% Mach: 0.6
% Vt : 670 ft/s
% AOA : 0.57 deg
% PA : 0.57 (pitch angle)
% FPA : 0 deg (flight path angle)
clear
bankstep = 0;
altstep = 1;
if bankstep == 1
 Frpole = 0.75;
 Lpole = 0.75;
 Ts = 0.1; % Time Step
end
if altstep == 1
 Frpole = 0.75;
 Lpole = 0.75;
 Ts = 0.1; % Time Step
 if Ts == 0.051
 Frpole = 0.85
 Lpole = 0.85
 end
end
Tstart = 0.0; % Start Time
Tstop = 30.0; % Stop Time
% Plant States, Inputs and Outputs
% x(1) = air speed, VT (ft/sec)
% x(2) = angle of attack, alpha (rad)
% x(3) = pitch angle, theta (rad)
% x(4) = pitch rate, Q (rad/sec)
% x(5) = engine thrust dynamics lag state, pow
% x(6) = angle of sideslip, beta (rad)
% x(7) = roll angle, phi (rad)
% x(8) = roll rate, P (rad/sec)
% x(9) = yaw rate, R (rad/sec)
% x(10) = altitude, h (feet)
```

```
% u(1) = throttle command 0.0 < u(1) < 1.0
% u(2) = elevator command in degrees
% u(3) = aileron command in degrees
% u(4) = rudder command in degrees
% y(1) = Az, Normal Acceleration (g's)
% y(2) = pitch rate, Q (deg/sec)
% y(3) = angle of attack, alpha (deg)
% y(4) = pitch angle, theta (deg)
% y(5) = air speed, VT (ft/sec)
% y(6) = Ay, Lateral Acceleration (g's)
% y(7) = roll rate, P (deg/sec)
% y(8) = yaw rate, R (deg/sec)
% y(9) = angle of sideslip, beta (deg)
% y(10) = roll angle, phi (deg)
% y(11) = altitude (feet)
%
% Get the trimed LTI state space matrices
%
[ascl,bscl,cscl,dreal,dscl,xeq,xeqall,ueq] = trim3;
eig(ascl)
along = ascl(1:5,1:5);
eig(along)
alat = ascl(6:9,6:9);
eig(alat)
%
% Establish the maximum and minimum control inputs for the model
% Note these are in deg and deg/sec, page 584 of Aircraft Control
% and Simulation
%
uabsmax = [0.765 25.0 21.5 0.5];
uabsmin = [0 -25.0 -21.5 -0.5];
umaxrate = [1e2 60.0 80.0 120.0];
uminrate = [-1e2 -60.0 -80.0 -120.0];
%
% The new output states
%
cscl = [cscl(10,:); % Bank angle in deg
 cscl(11,:); % Altitude in feet
 1 0 0 0 0 0 0 0 0 0; % Velocity (feet/sec)
 0 0 0 0 0 180/pi 0 0 0 0; % beta, Side Slip angle (deg)
 0 0 0 180/pi 0 0 0 0 0 0]; % theta dot, pitch rate (deg/sec)
```

```
dscl = [dscl(10,:);
 dscl(11,:);
 zeros(1,size(bscl,2));
 zeros(1,size(bscl,2));
 zeros(1,size(bscl,2))];
%
% Discritize the model
%
[A,B,C,D] = c2dm(ascl,bscl,cscl,dscl,Ts,'zoh');
%
% Check the rank of the observability and controllability
%
if rank(ctrb(A,B))~=size(A,1) | rank(obsv(A,C))~=size(A,1)
 disp('The rank of the Controllability Matrix is ');
 rank(ctrb(A,B))
 disp('The rank of the Observability Matrix is ');
 rank(obsv(A,C))
 disp('The size of the A matrix is');
 size(A)
end
%
%Set the continous time matrices
%
at = ascl;
bt = bscl;
ct = cscl;
dt = dscl;
%
%Determine the number of control inputs, outputs, and states
%
kappa = size(A,1); %estimator states
xi = size(B,2); %control inputs
eta = size(C,1); %system outputs
%
% Establish the prediction, optimization, and control horizons
%
p = 26; % State Prediction Horizon
q = 26; % Control Horizon
r = 5; % Optimization Horizon
if min([p q]) - 2*kappa <= r
 disp('The Optimization Horizon is within 2 times the number of states to min(p,q)');
 break
end
```

```
%
rho = max([p q]);
%
% Establish Z matrix weights
%
Zr = eye(xi);
Zl = eye(xi);
%
% Establish R optimization weights
%
if bankstep == 1
 Ry = eye(eta);
 Ry(1,1) = 1e2*Ry(1,1); % Bank Angle (deg)
 Ry(2,2) = 1e4*Ry(2,2); % Altitude (Ft)
 Ry(3,3) = 1e4*Ry(3,3); % Pert Velocity (ft/sec)
 Ry(4,4) = 1e2*Ry(4,4); % Side Slip Angle (deg)
 Ry(5,5) = 1e4*Ry(5,5); % Pitch Rate (deg/sec)
 Ru = eye(xi);
 Ru(1,1) = 1e-1*Ru(1,1);
 Ru(2,2) = 1e4*Ru(2,2);
 Ru(3,3) = 1e1*Ru(3,3);
 Ru(4,4) = 1e2*Ru(4,4);
end
if altstep == 1
 Ry = eye(eta);
 Ry(1,1) = 1e2*Ry(1,1); % Bank Angle (deg)
 Ry(2,2) = 1e4*Ry(2,2); % Altitude (Ft)
 Ry(3,3) = 1e4*Ry(3,3); % Pert Velocity (ft/sec)
 Ry(4,4) = 1e2*Ry(4,4); % Side Slip Angle (deg)
 Ry(5,5) = 1e4*Ry(5,5); % Pitch Rate (deg/sec)
 Ru = eye(xi);
 Ru(1,1) = 1e-1*Ru(1,1);
 Ru(2,2) = 1e4*Ru(2,2);
 Ru(3,3) = 1e1*Ru(3,3);
 Ru(4,4) = 1e2*Ru(4,4);
end
%
% Establish the constraint matrices: Ll, Mm, Nn
% The zeros are thrown in so that we do not want to constrain the future control
% inputs but rather only the next step i.e. the identity matrix. This
% is done because we will solve for the control inputs down the line
% but we will only implement the first step set of control inputs
%
```

```
Ll = [ -eye(xi) zeros(xi,xi*(q-1)) ;
 eye(xi) zeros(xi,xi*(q-1)) ];
Mm = [ -eye(xi) zeros(xi,xi*(q-1)) ;
 eye(xi) zeros(xi,xi*(q-1)) ];
Nn = [];
%
%Nn = [ -eye(kappa) zeros(kappa,kappa*(p-1)) ;
% eye(kappa) zeros(kappa,kappa*(p-1))];
%
%Establish physical constraint limits: l, m, n
%
l = 1*[(-uminrate*Ts) (umaxrate*Ts)]';
m = 1*[-(uabsmin-ueq) (uabsmax-ueq)]';
n = [];
%
% Go calculate the gains neccesary to make the system stable
%
[Fr,L,K]=gains3(A,B,C,D,Ts,Frpole,Lpole);
%
% Set up the controller matrices based upon the gains
%
[Ax,Bx,Cx,Dx,Ay,By,Cy,Dy,F0,F1,G,H]=contller(A,B,C,Fr,L,Zl,Zr);
%
% Establish the gain matirices for the non linear simulation
%
Kfeed = [K(1,1) K(1,2) K(1,6) K(1,7) K(1,3) 0 K(1,8) K(1,4) K(1,9) 0 0 K(1,10) K(1,5);
 K(2,1) K(2,2) K(2,6) K(2,7) K(2,3) 0 K(2,8) K(2,4) K(2,9) 0 0 K(2,10) K(2,5);
 K(3,1) K(3,2) K(3,6) K(3,7) K(3,3) 0 K(3,8) K(3,4) K(3,9) 0 0 K(3,10) K(3,5);
 K(4,1) K(4,2) K(4,6) K(4,7) K(4,3) 0 K(4,8) K(4,4) K(4,9) 0 0 K(4,10) K(4,5)];
Kfeed = zeros(4,13);
P = 180/pi;
%
% Non linear Matrix Gains, Used for picking off outputs
%
Krc = [0 0 0 0 0 0 0 0 0 0 0 1 0];
Kh = Krc;
Kphi = [0 0 0 P 0 0 0 0 0 0 0 0 0];
Kvel = [1 0 0 0 0 0 0 0 0 0 0 0 0];
Kbeta = [0 0 P 0 0 0 0 0 0 0 0 0 0];
Kthetd = [0 0 0 0 0 0 0 P 0 0 0 0 0];
```

```
Klook1 = [1 0 0 0 0 0 0 0 0 0 0 0 0 0; % Vel
 0 P 0 0 0 0 0 0 0 0 0 0 0 0; % alpha (deg)
 0 0 0 0 P 0 0 0 0 0 0 0 0 0; % Pitch (deg)
 0 0 0 0 0 0 0 P 0 0 0 0 0 0; % Q Pitch Rate (deg)
 0 0 0 0 0 0 0 0 0 0 0 0 1 0; % altitude (feet)
 0 0 0 0 0 0 0 0 0 0 0 0 0 1]; % engine thrust
Klook2 = [0 0 P 0 0 0 0 0 0 0 0 0 0 0; % beta (deg)
 0 0 0 P 0 0 0 0 0 0 0 0 0 0; % phi (deg)
 0 0 0 0 0 0 P 0 0 0 0 0 0 0; % P. Roll Rate (deg)
 0 0 0 0 0 P 0 0 0 0 0 0 0 0; % Yaw (deg)
 0 0 0 0 0 0 0 0 P 0 0 0 0 0]; % R, yaw rate (deg)
%
% Linear Matrix Gains, Used for picking off outputs
%
KvelL = [1 0 0 0 0 0 0 0 0 0 0 0];
KbetaL =[0 0 0 0 0 P 0 0 0 0 0 0];
KphiL = [0 0 0 0 0 0 P 0 0 0 0 0];
KqdotL =[0 0 0 P 0 0 0 0 0 0 0 0];
KhL = [0 0 0 0 0 0 0 0 0 0 0 1];
%
% Set up the prediction matrices
%
[cF,cG,cGinf,cH,cFu,cGu,cGuinf,cHu,cFdel,cGdel,cGdelinf,cHdel,cIdel] = ...
predmat(kappa,xi,eta,p,q,r,rho,F0,F1,G,H,Fr,Zr);
%
% Set up the contraints
%
[S,T,cD,cE] = qpconst(C,Ry,Ru,cF,cG,cGinf,cH,cFdel,cGdel,cHdel, ...
 cGdelinf,cFu,cGu,cHu,cGuinf,Ll,Mm,Nn,p,q,r,rho);
%
```

```
% Establish the initial values
%
xhat0 = 0.0*ones(1,kappa);
x0 = zeros(1,kappa);
u0 = zeros(1,xi);
%
% Set the number of x dot terms coming out of the xd routine
% for the F-16 Model
%
kapf16 = 13;
x0f16 = zeros(1,kapf16);
%
% Save the Matrices to be used later
%
save mats1 A B C D Fr L Zr T S cD cE l m n ueq
ailu = [];
aill = [];
save ail ailu aill
%
% Start the Simulink diagram for the non-linear simulation
%
nlinsim
```

A.5 Gains3.m

```
function [Fr,L,K]=gains3(A,B,C,D,Ts,zstgoal,zesgoal);
%
% function [Fr,L,K]=gains3(A,B,C,D,Ts,zstates,zestim)
%
% Function designed to make the poles of the discrete
% system A + B*Fr less than zstates and the poles of the
% discrete system A + L*C less than zestim
%
% Fr, Discrete Gains
% K, Continuous Gains
%
% Convert the system to a continous models
%
[Ac,Bc,Cc,Dc] = d2cm(A,B,C,D,Ts,'zoh');
%
% Get the number of states
kappa = size(A,1);
per = 0.95;
red = 0.98;
zstate = zstgoal;
zestim = zesgoal;
zst = ones(kappa,1);
cnt = 0;
%
% Place the discrete poles of the state feedback
%
while max(abs(zst)) > zstgoal & cnt<10
 pst = (log(zstate)/Ts):...
 (log(per*zstate)/Ts - log(zstate)/Ts)/(kappa-1):...
 (log(per*zstate)/Ts);
% pst(10) = log(0.1)/Ts
 Frc = -place(Ac,Bc,pst);
 Fr = -place(A,B,exp(eig(Ac+Bc*Frc).*Ts));
 zst = eig(A+B*Fr);
 if max(abs(zst)) > zstgoal
 cnt = cnt + 1;
 zstate = red * zstate;
 else
 cnt = 12;
 end
end
pes = ones(kappa,1);
cnt = 0;
```

```
%
% Place the discrete poles of the estimator
%
while max(abs(pes)) > zesgoal & cnt<10
 pes = (log(zestim)/Ts):...
 (log(per*zestim)/Ts - log(zestim)/Ts)/(kappa-1):...
 (log(per*zestim)/Ts);
% pes(10) = log(0.5)/Ts
 Lc = -(place(Ac',Cc',pes))';
 L = -(place(A',C',exp(eig(Ac+Lc*Cc).*Ts)))';
 zes = eig(A+L*C);
 if max(abs(zes)) > zesgoal
 cnt = cnt + 1;
 zestim = red * zestim;
 else
 cnt = 12;
 end
end
%
% Place the continuous poles of the state feedback
%
pole1 = -2;
pole2 = -3;
conpoles = pole1:(pole2-pole1)/(kappa-1):pole2;
K = -place(Ac,Bc,conpoles);
disp('Eigen Values of A + B*Fr');
eig(A+B*Fr)
disp('Eigen Values of A + L*C');
eig(A+L*C)
disp('Eigen Values of Ac + Bc*K');
eig(Ac+Bc*K)
```

A.6 Optimize.m

```
function [v] = optimize(cin);
%
% function [v] = optimize(cin);
%
% This m-file will run a constrained quadratic optimization to find the
% quasi-reference v(k) that will be input to the plant.
% INPUTS
%
% cin - input vector that is multiplexed by simulink file
% cin(1) = t, time
% cin(2) = Ts, time step
% cin(3) = kappa, Number of states
% cin(4) = xi, Number of control inputs
% cin(5) = eta, Number of outputs
% cin(6) = kapf16, Number of subf16 states for F-16 Model
% cin(7) = p, State Prediction Horizon
% cin(8) = q, Control Horizon
% cin(9) = r, Optimization Horizon
% cin(10:xi+9) = u, control inputs
% cin(10+xi:xi+kapf16+9) = x0, Initial F-16 Model states
%
% Written by Chris Shearer, Aug 97
%
t = cin(1) % Time
Ts = cin(2); % Time Step
kappa = cin(3); % Number of Estimator States
xi = cin(4); % Number of Control Inputs
eta = cin(5); % Number of System Outputs
kapf16 = cin(6); % Number of F-16 Model States
p = cin(7); % State Prediction Horizon
q = cin(8); % Control Horizon
r = cin(9); % Optimization Horizon
rho = max([p q]);
%
% Pick apart the optimizer input from Simulink
% based upon the order of the multiplexier (Mux)
%
xhat = cin(10:kappa+9);
y = cin(kappa+10:kappa+eta+9);
u = cin(kappa+eta+10:kappa+eta+xi+9);
%
```

```
% Now based upon the current value of y go get
% A, B, C, D Discretized State Space Model
% Fr, State Gains
% L, Estimator Gains
% Zr, Optimized v input gains (typically Identity)
% T and S, matrices used in qp probelm
% cD and cE, contraint equations for qp problem
%
[A,B,C,D,Fr,L,Zr,T,S,cD,cE,l,m,n,ueq] = matget(y);
%
% Now determine the current trajectory based upon
% the current time, prediction and optimization horizions
%
[s,sinf] = trajpnt(t,Ts,p,r,0);
%
% This is for addaptive control
%
% First I get absolute max and minimum control inputs
% Then apply scalling equations (ail = -0.0600 * ail/rollrate)
% add a 1.25% buffer
% then apply new max an mins
%
if 1 == 2
 bank = y(1);
 bankdotp = (s(1+eta)-s(1))/Ts;
 bankdote = (s(1+eta)-bank)/Ts;
 altdot = (s(2+eta) - s(2))/Ts;
 altddotp = (s(2+2*eta) - 2*s(2+eta) + s(2))/Ts^2;
 alt = y(2);
 altddote = (s(2+2*eta) - 2*s(2+eta) + alt)/Ts^2;
 umax = m(5:8)' + ueq;
 if bankdotp == 0 & altdot == 0
 uail = (-0.06/2)*bankdote;
 buffa = (0.5/100)*umax(3);
 elseif altdot == 0
 uail = -0.06*bankdotp;
 buffa = (0.5/100)*umax(3);
 else
 uail = -0.06*bankdotp + (-0.06/2)*bankdote
 buffa = (0.5/100)*umax(3);
 end
```

```
 uailmax = uail + buffa;
 uailmin = uail - buffa;
 load ail
 ailu = [ailu uailmax];
 aill = [aill uailmin];
 save ail ailu aill
 m(3) = -uailmin;
 m(7) = uailmax;
end
if 1 == 2
 uel1 = -altddotp/45;
 uel = uel1;
 uelmax = uel + (2/100)*umax(2);
 uelmin = uel - (2/100)*umax(2);
 m(2) = -uelmin;
 m(6) = uelmax;
end
[vinf] = vinfy(A,B,C,Fr,Zr,sinf,r,rho);
Tstar = [ xhat' y' vinf' s' ]*T;
[x,lambda,how] = qp(2*S,Tstar,cD,cE*[xhat' vinf' y' u' l' m' n']');
% Unconstrained
%[x,lambda,how] = qp(2*S,Tstar,0*cD,cE*[xhat' vinf' y' u' l' m' n']');
v = x(1:xi);
```

APPENDIX B - Operating Envelope and F-16 Planform and Longitudinal Views

B.1 F-16 Planform and Longitudinal Views

Removed for Distribution Purposes

B.2 F-16 Operating Envelope/Turn Performance - Sea Level

Removed for Distribution Purposes

APPENDIX C - LTI Models

C.1 HARV

$$A = \begin{bmatrix} -7.5000e-2 & -2.4050e+1 & 0 & -3.2160e+1 \\ -8.8250e-4 & -1.9590e-1 & 9.8960e-1 & 0 \\ -1.5000e-4 & -1.4540e-1 & -1.8770e-1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} -2.298e-2 & 0 & -7.286e-2 & 3.930e-2 & -4.114e-2 & 1.600e-1 \\ -1.800e-4 & -1.300e-4 & -4.300e-4 & -4.700e-5 & -3.000e-4 & -3.000e-4 \\ -6.700e-3 & -7.000e-4 & -1.200e-2 & -5.700e-4 & 7.000e-4 & 4.700e-4 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$D = 0$$

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{\pi}{180} & 0 & 0 \\ 0 & 0 & \frac{\pi}{180} & 0 \\ 0 & 0 & 0 & \frac{\pi}{180} \end{bmatrix}$$

$$S = \begin{bmatrix} 20 & 0 & 0 & 0 & 0 & 0 \\ 0 & 50 & 0 & 0 & 0 & 0 \\ 0 & 0 & 34 & 0 & 0 & 0 \\ 0 & 0 & 0 & 30 & 0 & 0 \\ 0 & 0 & 0 & 0 & 60 & 0 \\ 0 & 0 & 0 & 0 & 0 & 27 \end{bmatrix}$$

$$R = \begin{bmatrix} 8 & 0 & 0 \\ 0 & \frac{\pi}{180} & 0 \\ 0 & 0 & \frac{\pi}{180} \end{bmatrix}$$

The A, B, C, and D matrices that were used in the simulations were scaled using the following relationship

$$A_{scl} = T^{-1}AT$$

$$B_{scl} = T^{-1}AS$$

$$C_{scl} = R^{-1}AT$$

$$D_{scl} = R^{-1}AS$$

C.2 F-16

Listed below are the LTI state space matrices used in the linear simulations

$$A = \begin{bmatrix} -2.53e-02 & 3.94e+01 & -3.22e+01 & 2.41e-01 & 4.28e-01 & -1.45e-06 \\ -1.42e-04 & -1.34e+00 & -2.68e-06 & 9.08e-01 & -7.12e-06 & 2.62e-06 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6.35e-13 & -4.40e+00 & 0 & -1.83e+00 & 0 & 1.13e-06 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 4.45e-14 & 0 & 3.38e-13 & 0 & 0 & -4.28e-01 \\ 0 & 0 & 0 & -7.00e-12 & 0 & 0 \\ -2.57e-13 & -1.38e-11 & 0 & 2.63e-04 & 0 & -4.74e+01 \\ -4.01e-15 & 2.50e-12 & 0 & 2.54e-03 & 0 & 1.67e+01 \\ -3.65e-14 & -6.70e+02 & 6.70e+02 & 0 & 0 & 4.03e-07 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 8.43e-05 \\ 0 & 0 & 0 & 1.40e-06 \\ 0 & 0 & 6.27e-10 & 0 \\ 0 & -8.80e-06 & -2.86e-03 & -6.16e-15 \\ 0 & 0 & 0 & 0 \\ 4.80e-02 & 9.90e-03 & -9.93e-01 & -7.06e-20 \\ 0 & 1 & 1.12e-02 & 0 \\ 0 & -4.97e+00 & 7.15e-01 & 2.50e-15 \\ 0 & 1.94e-03 & -6.61e-01 & 3.89e-17 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 3.70e-1 & 0 & 0 \\ 0 & -2.8463e-03 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -3.2464e-01 & 0 & 0 \\ 6.49e+1 & 0 & 0 & 0 \\ 0 & 0 & 3.92e-04 & 1.07e-03 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -1.29e+00 & 2.38e-01 \\ 0 & 0 & -5.91e-02 & -1.14e-01 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & \frac{180}{\pi} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{180}{\pi} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{180}{\pi} & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$D = 0$$

APPENDIX D - Extra Simulation Results

D.1 Combined 35% C.G. Location, Linear vs Nonlinear Comparison

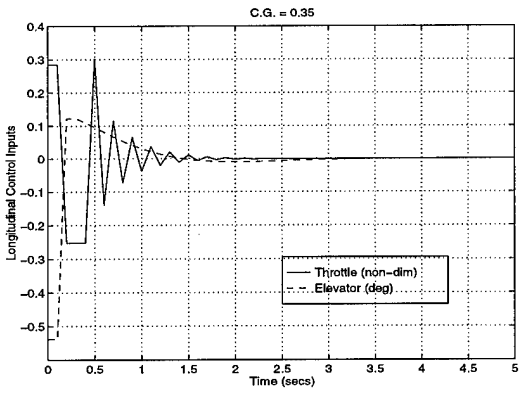

Figure 94. Linear and Nonlinear Comparison - Longitudinal Control Inputs

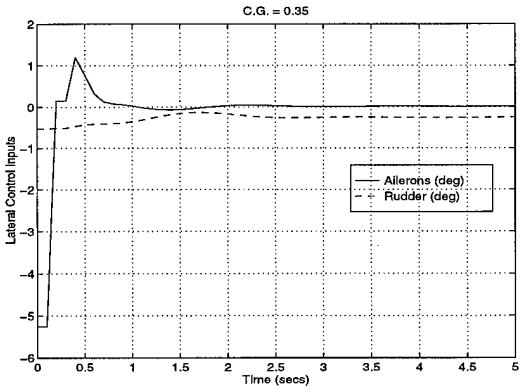

Figure 95. Linear and Nonlinear Comparison - Lateral Control Inputs

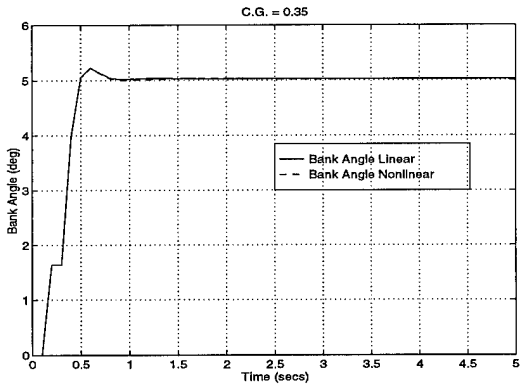

Figure 96. Linear and Nonlinear Comparison - Bank Angle

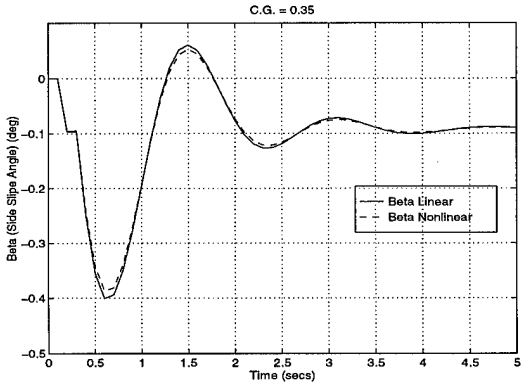

Figure 97. Linear and Nonlinear Comparison - Side Slip Angle

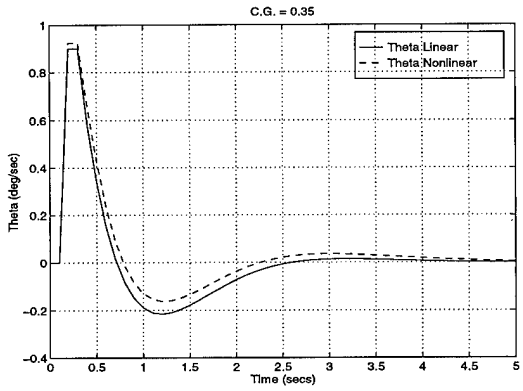

Figure 98. Linear and Nonlinear Comparison - Pitch Rate

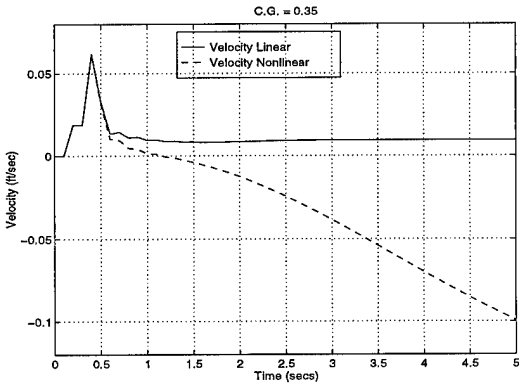

Figure 99. Linear and Nonlinear Comparison - Velocity

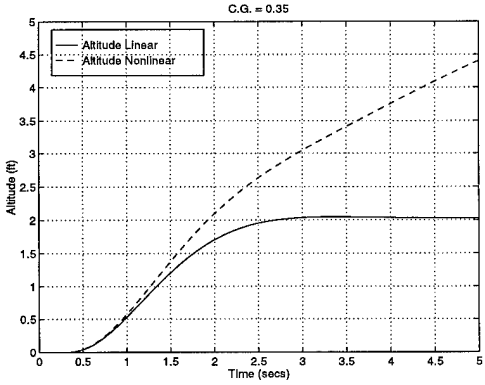

Figure 100. Linear and Nonlinear Comparison - Altitude

D.2 Case 3

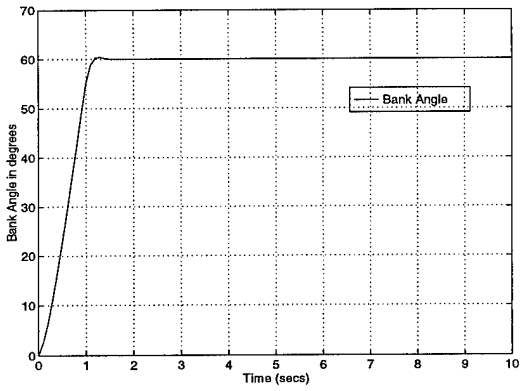

Figure 101. Case 3 - Bank Angle Output

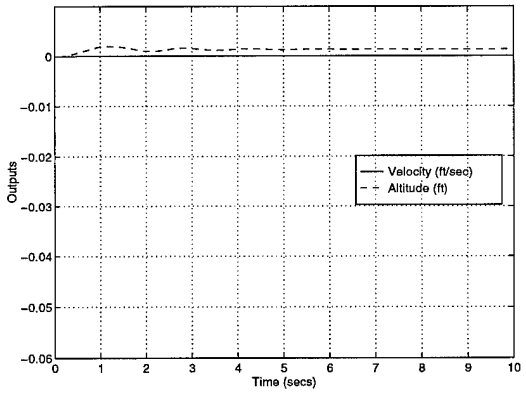

Figure 102. Case 3 - Velocity and Altitude Outputs

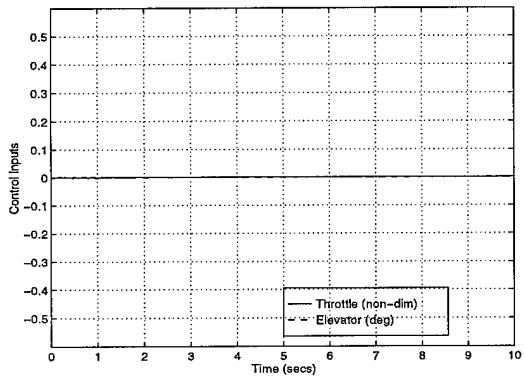

Figure 103. Case 3 - Longitudinal Controls

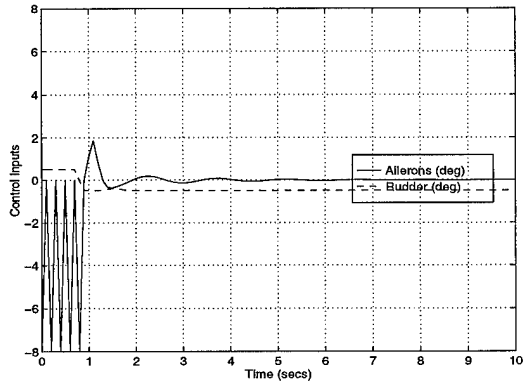

Figure 104. Case 3 - Lateral Controls

D.3 Case 4

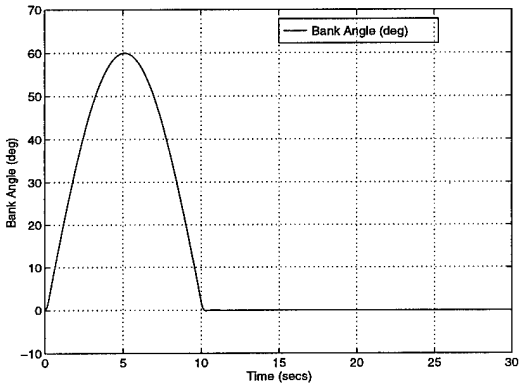

Figure 105. Case 4 - Bank Angle Output

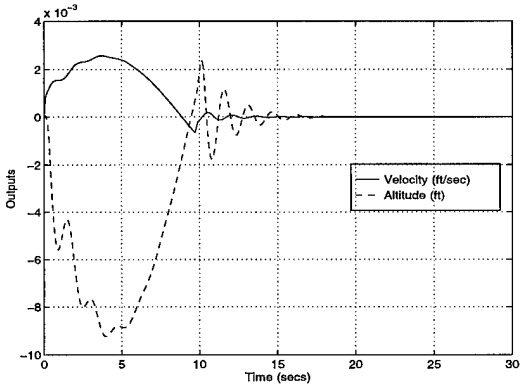

Figure 106. Case 4 - Velocity and Altitude Outputs

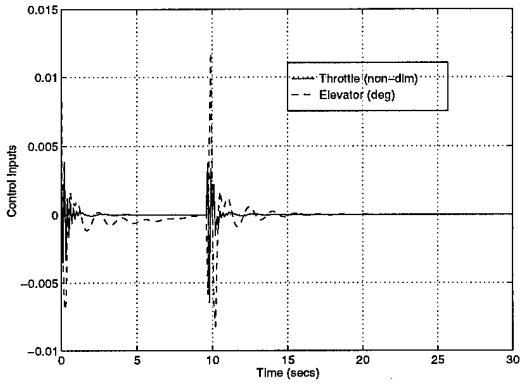

Figure 107. Case 4 - Longitudinal Controls

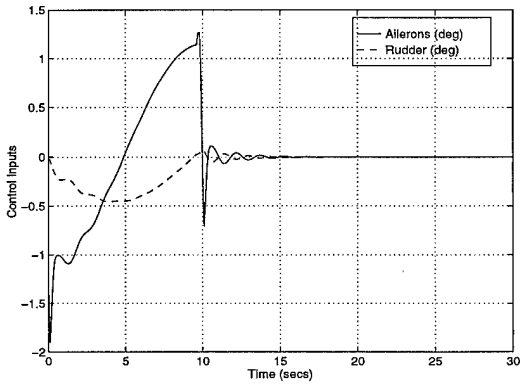

Figure 108. Case 4 - Lateral Controls

D.4 Case 5

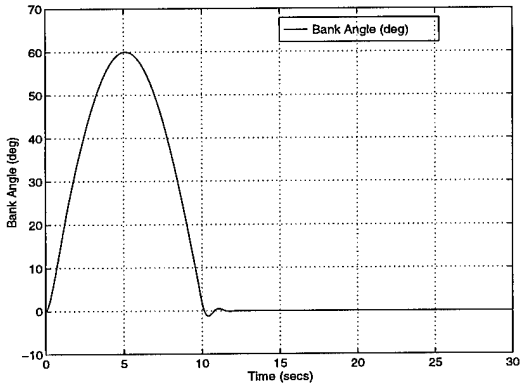

Figure 109. Case 5 - Bank Angle Output

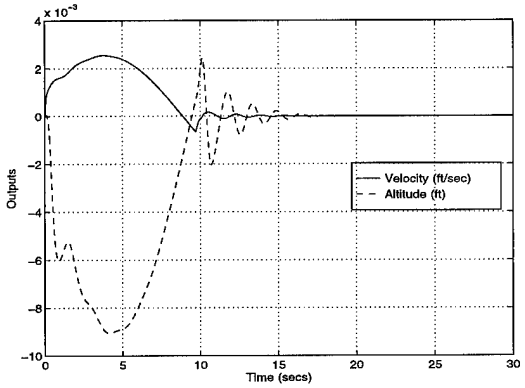

Figure 110. Case 5 - Velocity and Altitude Outputs

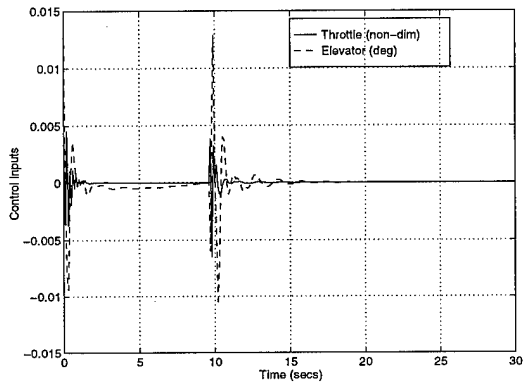

Figure 111. Case 5 - Longitudinal Controls

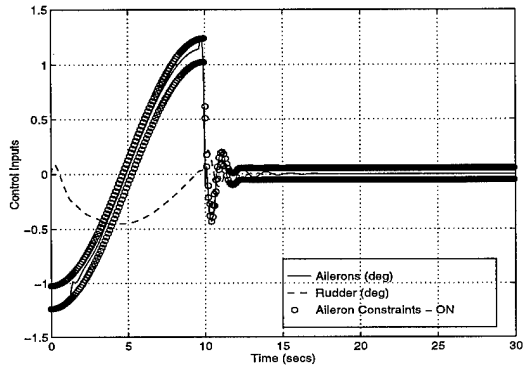

Figure 112. Case 5 - Lateral Controls

D.5 Case 14

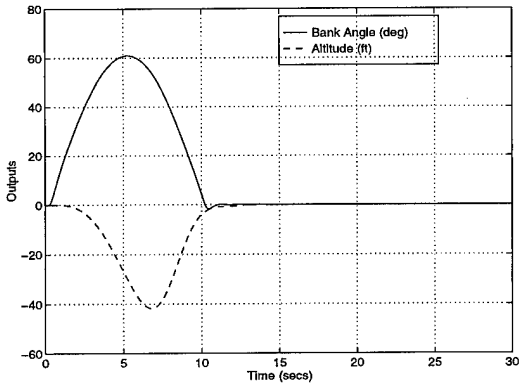

Figure 113. Case 14 - Bank Angle and Altitude

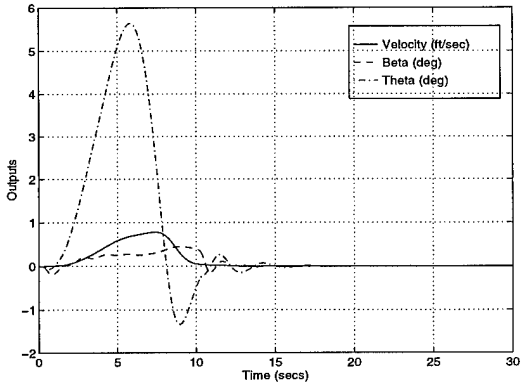

Figure 114. Case 14 - Velocity, Beta and Theta

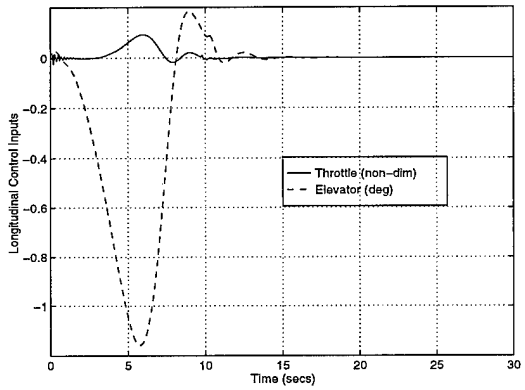

Figure 115. Case 14 - Longitudinal Control Inputs

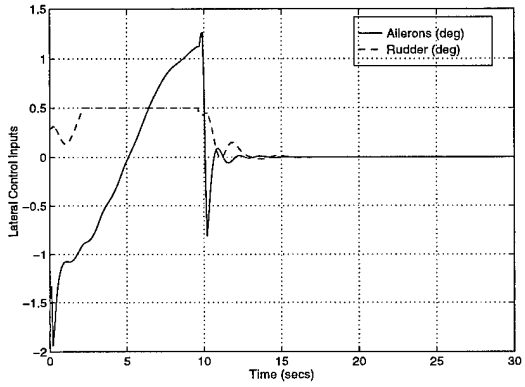

Figure 116. Case 14 - Lateral Control Inputs

D.6 Case 15

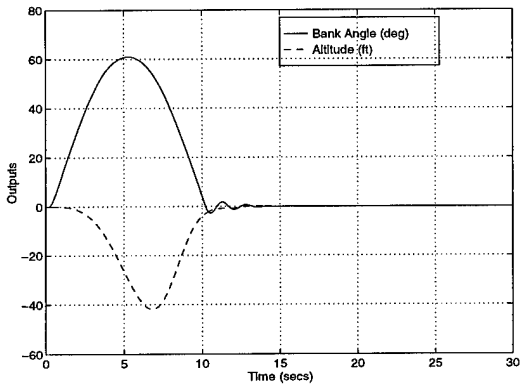

Figure 117. Case 15 - Bank Angle and Altitude

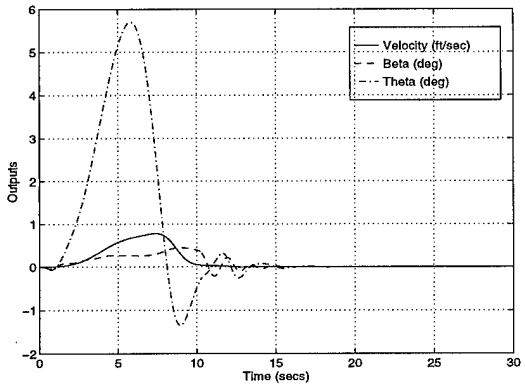

Figure 118. Case 15 - Velocity, Beta and Theta

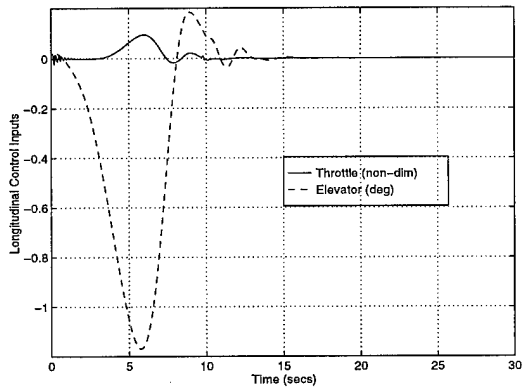

Figure 119. Case 15 - Longitudinal Control Inputs

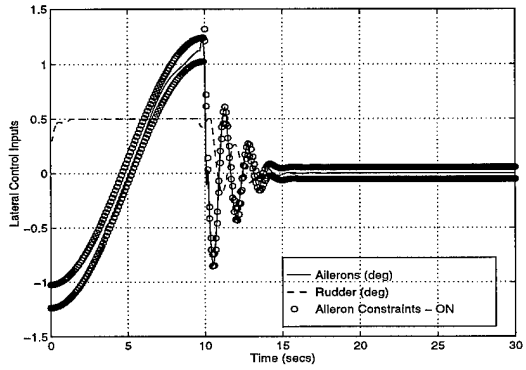

Figure 120. Case 15 - Lateral Control Inputs

D.7 Case 16

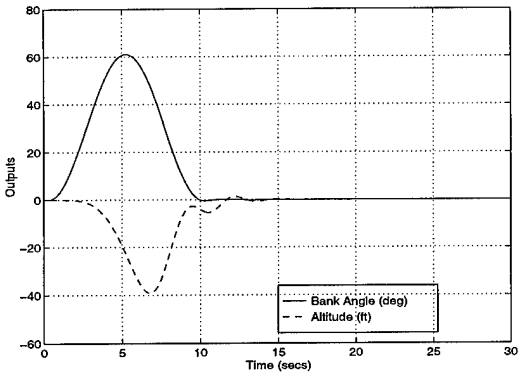

Figure 121. Case 16 - Bank Angle and Altitude

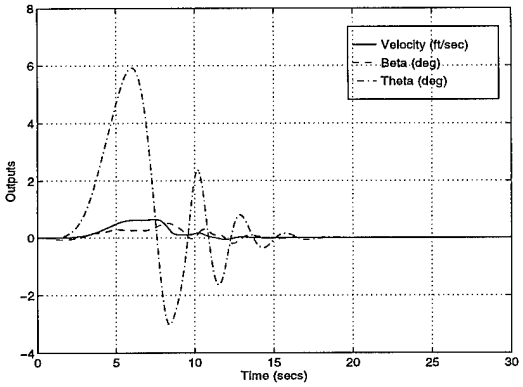

Figure 122. Case 16 - Velocity, Beta and Theta

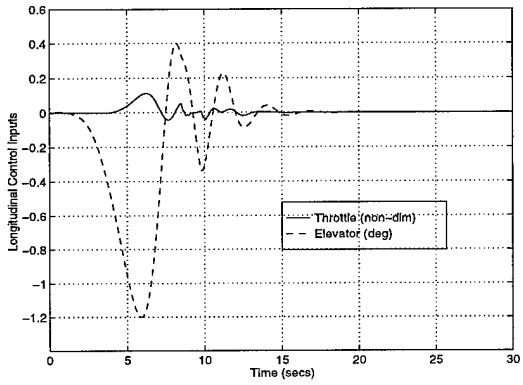

Figure 123. Case 16 - Longitudinal Control Inputs

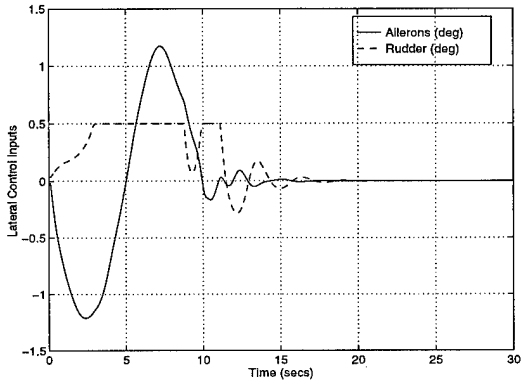

Figure 124. Case 16 - Lateral Control Inputs

D.8 Case 19

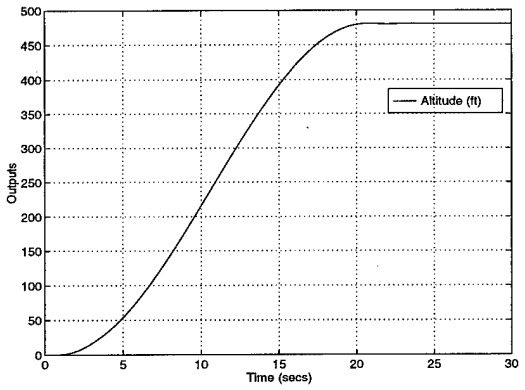

Figure 125. Case 19 - Altitude Output

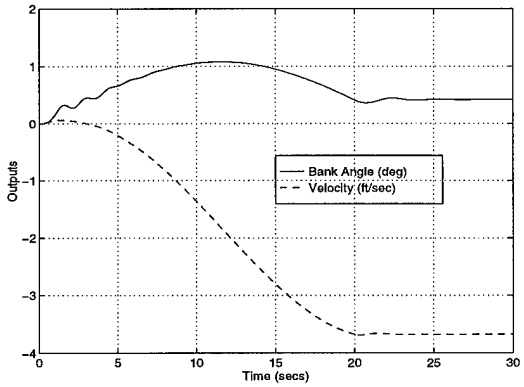

Figure 126. Case 19 - Velocity and Bank Angle Outputs

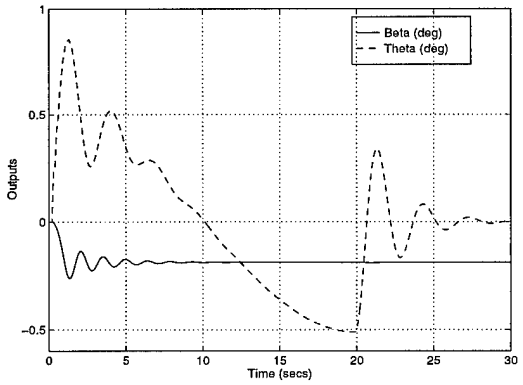

Figure 127. Case 19 - Beta and Theta Outputs

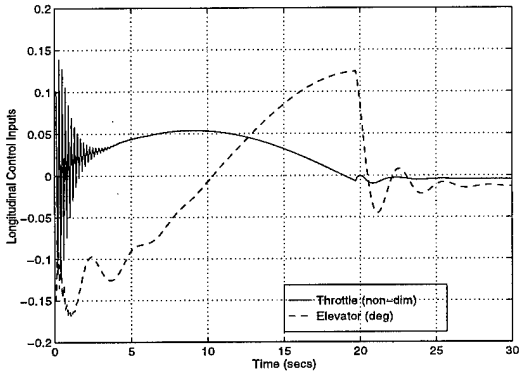

Figure 128. Case 19 - Longitudinal Control Inputs

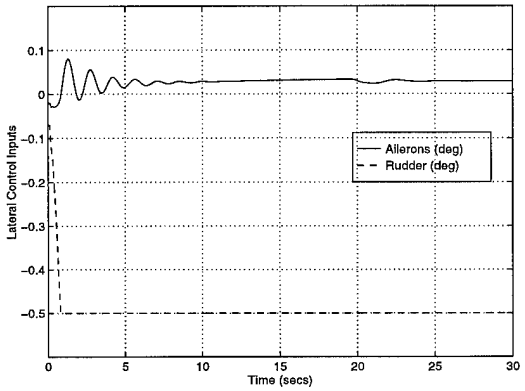

Figure 129. Case 19 - Lateral Control Inputs

D.9 Case 21

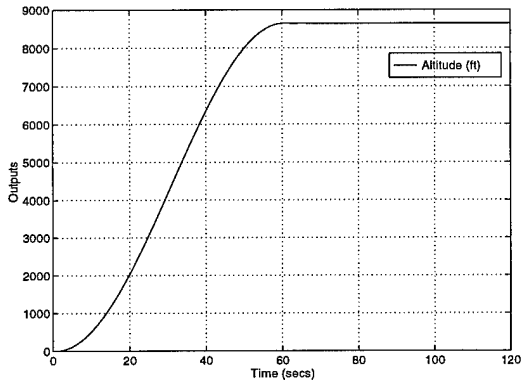

Figure 130. Case 21 - Altitude Output

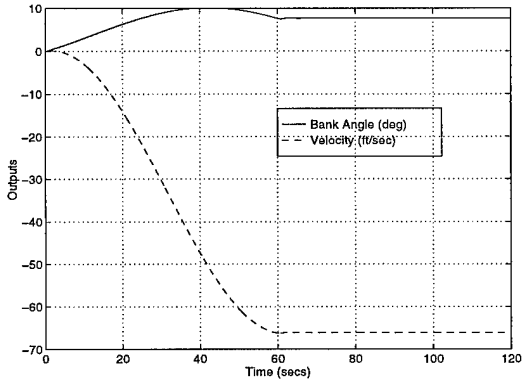

Figure 131. Case 21 - Velocity and Bank Angle Outputs

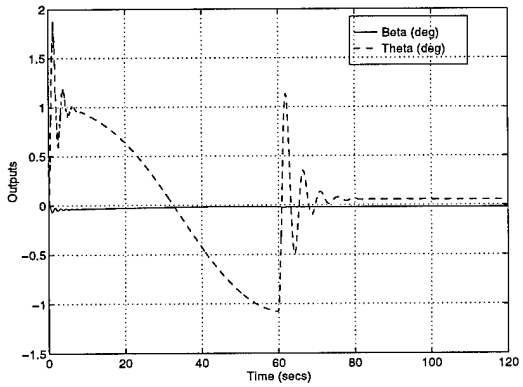

Figure 132. Case 21 - Beta and Theta Outputs

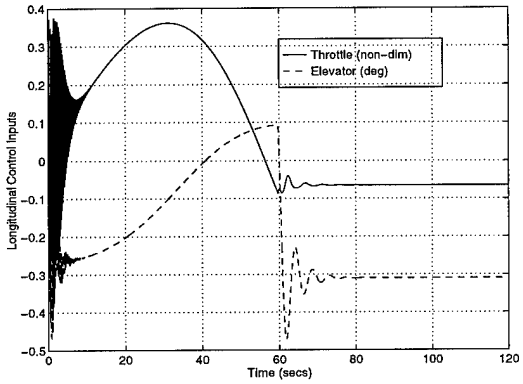

Figure 133. Case 21 - Longitudinal Inputs

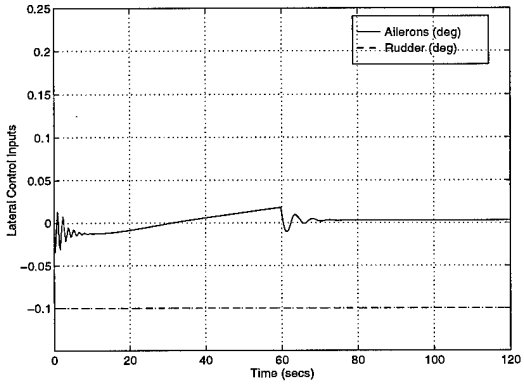

Figure 134. Case 21 - Lateral Inputs

D.10 Case 22

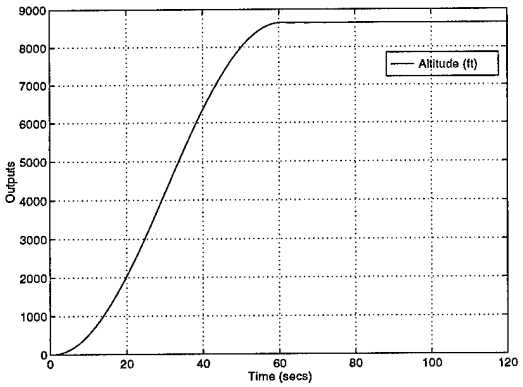

Figure 135. Case 22 - Altitude Output

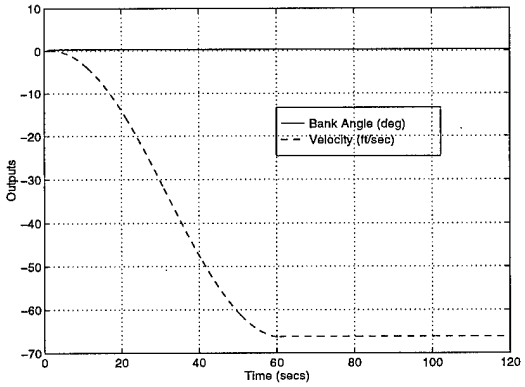

Figure 136. Case 22 - Velocity and Bank Angle Outputs

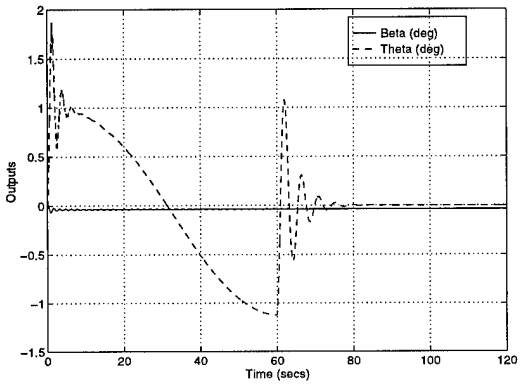

Figure 137. Case 22 - Beta and Theta Outputs

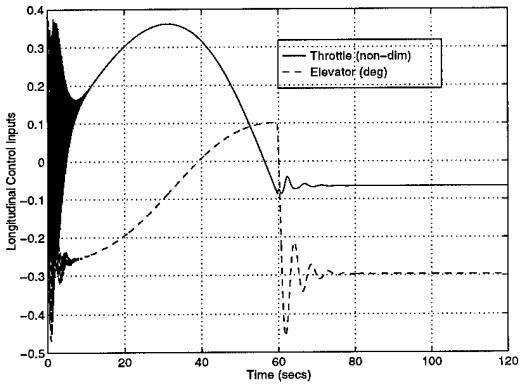

Figure 138. Case 22 - Longitudinal Inputs

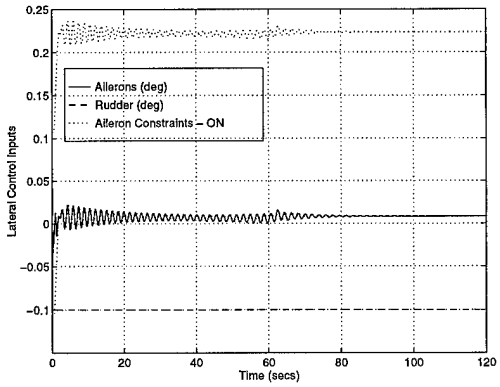

Figure 139. Case 22 - Lateral Inputs

D.11 Case 24

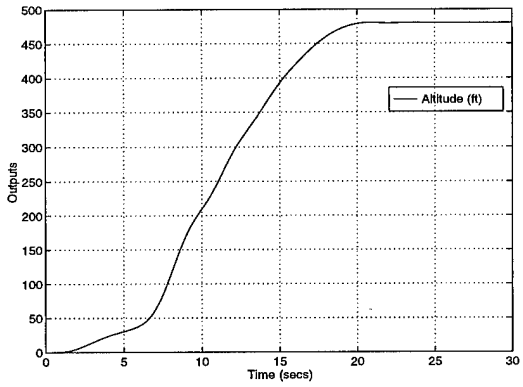

Figure 140. Case 24 - Altitude Output

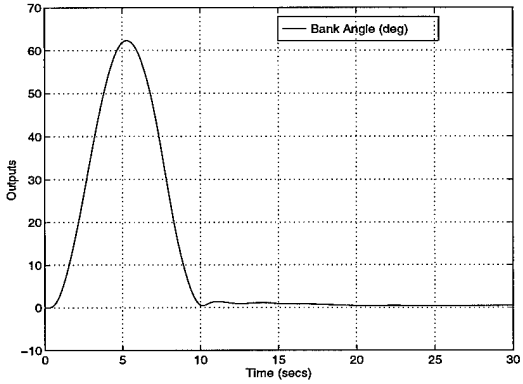

Figure 141. Case 24 - Bank Angle Output

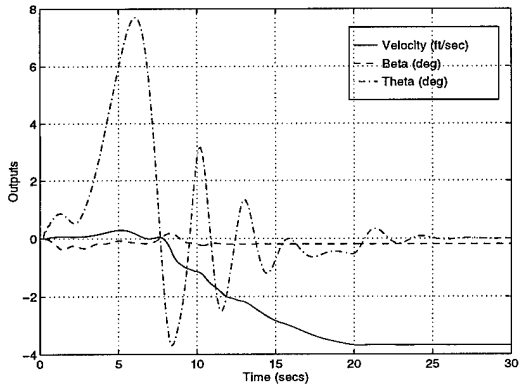

Figure 142. Case 24 - Velocity, Beta and Theta Outputs

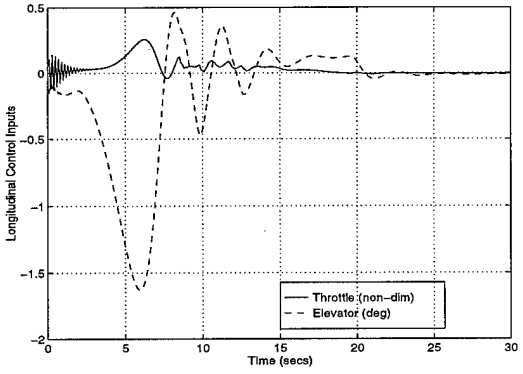

Figure 143. Case 24 - Longitudinal Controls

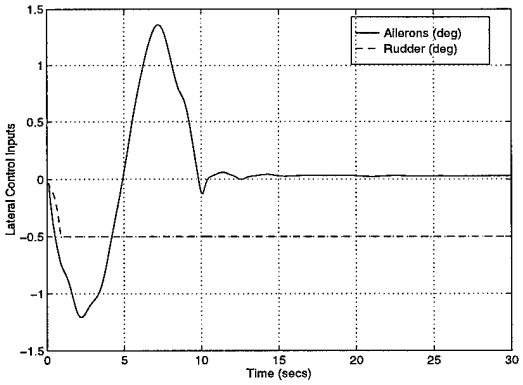

Figure 144. Case 24 - Lateral Controls

D.12 Case 26

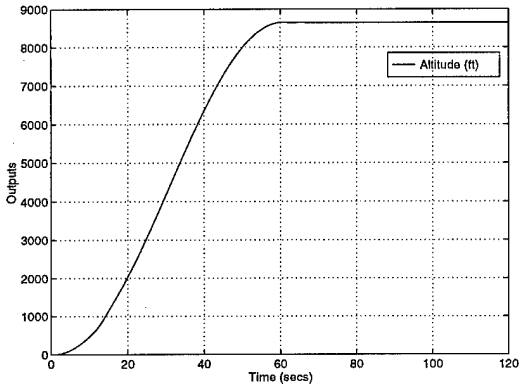

Figure 145. Case 26 - Altitude Output

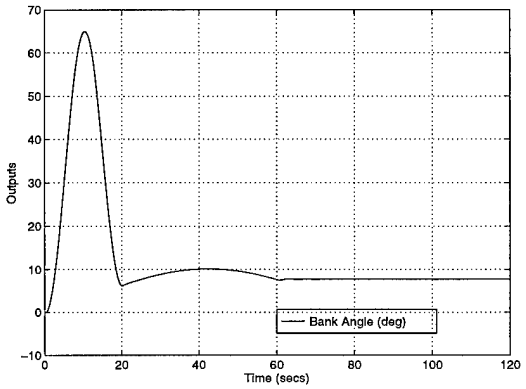

Figure 146. Case 26 - Bank Angle Output

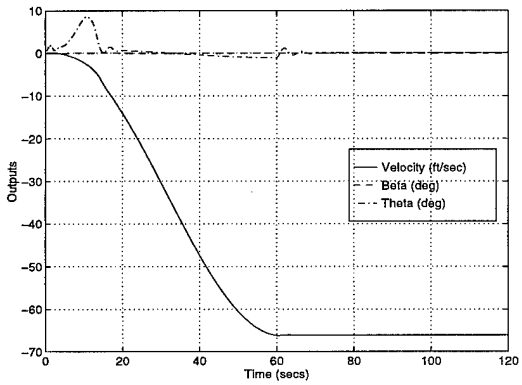

Figure 147. Case 26 - Velocity, Beta and Theta Outputs

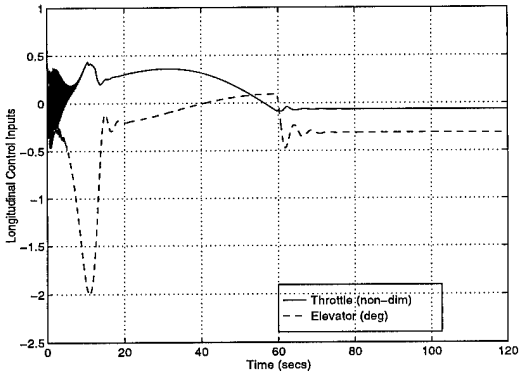

Figure 148. Case 26 - Longitudinal Controls

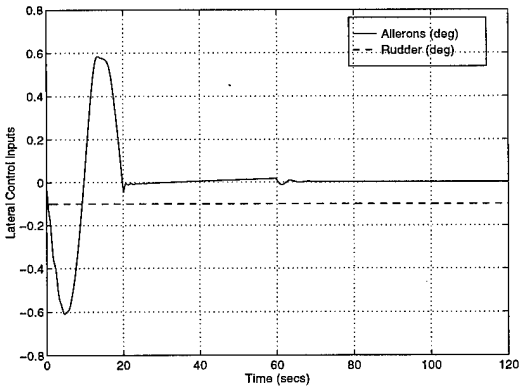

Figure 149. Case 26 - Lateral Controls

Bibliography

[1] *T.O. 1 F-16C-1, Flight Manual*, February 1995.

[2] Chapa, Capt Mike and Capt Barry St. Germain. "JACOB.M." MATLAB Function Call, Winter 1997.

[3] Chapa, Capt Mike and Capt Barry St. Germain. "TRIMMER.M." MATLAB Function Call, Winter 1997.

[4] Ebdon, Derek W. *Model Predicitive Control of Aerospace Systems*. MS thesis, Air Force Institute of Technology, December 1996.

[5] Franklin, Gene F., et al. *Feedback Control of Dynamic Systems*. Addison-WesleyPublishing Company, 1994.

[6] Heise, Sharon Ann. *Multivariable Constrained Model Predicitive Control*. PhD dissertation, Pembroke College, University of Cambridge, December 1994.

[7] Kouvaritakis, B., et al. "Stable Generalized Predictive Control: An Algorithm with Guaranteed Stability," *IEE Proceedings-D*, *139* 4, 349–362 1992.

[8] Liebst, Dr. Brad, et al. "MECH 628." Instructor: Dr. Brad Liebst, Winter 1997.

[9] Nelson, Robert C. *Flight Stability and Automatic Control*. Mc-Graw Hill Book Company, 1989.

[10] Pendleton, E., et al. "Application of Active Flexible Wing Technology to the Agile Falcon," *Journal of Aircraft*, *29* 3, 444–451 1992.

[11] Singh, Sahjendta N., et al. "Nonlinear Predicitive Control of Feedback Linearizable Systems and Flight Control System Design," *Journal of Guidance, Control, and Dynamics*, *Vol. 18, No. 5* 1023–1028 1995.

[12] Stevens, Brian L. and Frank L. Lewis. *Aircraft Control and Simulation*. John Wiley and Sons, Inc., 1992.

[13] Stevens, Dr. B. L. and Dr. Brad Liebst. "SUBF16A.M." MATLAB Function Call originally written by Dr. B.L Stevens and transcribed by Dr. Brad Liebst, 1996.

[14] Zhou, Kemin, et al. *Robust and Optimal Control*. Prenticd Hall, 1996.

Vita

Capt Christopher Shearer was born in McAllen, Texas on April 21st 1969. At an early age he became fascinated with airplanes and dreamed of joining the Air Force. After graduating from Millard North High School in Omaha Nebraska, he attended and graduated from Texas A&M University with a bachelors degree in Aerospace Engineering and a commission as a Second Lieutenant in the Air Force in May 1992. His first assignment was to the Training System Product Group (TSPG), Wright Patterson AFB in March 1993. During his two years at the TSPG he worked in the front office, was a program engineer on the T-3A program, and worked as a program manager on a visual upgrade to the B-1 and B-2 simulators. While at the TSPG he married his wife Lt. Heather L. Shearer. Following the TSPG Capt Shearer was assigned to the Aeroelasticity Branch of the Flight Dynamics Directorate at Wright Laboratory where he worked issues on the Active Aeroelastic Wing program. He was then assigned to AFIT to earn his Masters degree in Aeronautical Engineering. Following graduation he was assigned to Phillips Lab, Kirtland AFB New Mexico.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	December 1997	Master's Thesis

4. TITLE AND SUBTITLE
CONSTRAINED MODEL PREDICTIVE CONTROL OF A NONLINEAR AEROSPACE SYSTEM

5. FUNDING NUMBERS

6. AUTHOR(S)
Christopher M. Shearer, Captain, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology
2750 P Street
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GAE/ENY/97D-05

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
WL/FIGC BLDG 146
2210 EIGTH STREET SUITE 21
WPAFB OH 45433-7521

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Recent research efforts have applied the receding horizon Model Predicitive Control (MPC) strategy to linearized high performance aerospace systems. The research contained in this thesis used thes recent results in order to apply the MPC strategy to a nonlinear high performance aerospace system, speciffically an F-16 fighter aircraft model. The model was commanded to follow dynamic trajectories of roll angle and altitude. Further, adaptive constraint techniques were used to improve system tracking.

To accomplish these tasks, code and block diagrams were generated using the commercial software packages of Matlab and Simulink. Numerous simulations were conducted with the goal of acheiving realistic aircraft performance. In many cases, to improve system tracking and reduce control input oscillations, rigid mathematical constraints previously used in the MPC strategy were relaxed.

14. SUBJECT TERMS	15. NUMBER OF PAGES
MPC, Predictive Control, Nonlinear Simulation	146
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL